①

Automatic Concept Formation in a Rich Input Domain

Michael Lebowitz

Columbia University

for

Contracting Officer's Representative
Judith Orasanu

ARI Scientific Coordination Office, London
Milton S. Katz, Chief

Basic Research Laboratory
Michael Kaplan, Director

U. S. Army

Research Institute for the Behavioral and Social Sciences

June 1988

U. S. ARMY RESEARCH INSTITUTE

FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency under the Jurisdiction of the

Deputy Chief of Staff for Personnel

EDGAR M. JOHNSON
Technical Director

L. NEALE COSBY
Colonel, IN
Commander

Research accomplished under contract
for the Department of the Army

Columbia University

Technical review by

Tracye Julien

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified		1b. RESTRICTIVE MARKINGS n/a			
2a. SECURITY CLASSIFICATION AUTHORITY - -		3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.			
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE - -					
4. PERFORMING ORGANIZATION REPORT NUMBER(S) - -		5. MONITORING ORGANIZATION REPORT NUMBER(S) ARI Research Note 88-54			
6a. NAME OF PERFORMING ORGANIZATION Columbia University	6b. OFFICE SYMBOL (If applicable) - -	7a. NAME OF MONITORING ORGANIZATION Army Research Institute for the Behavioral and Social Sciences			
6c. ADDRESS (City, State, and ZIP Code) 715 Broadway, 5th Floor New York, NY 10003-6896		7b. ADDRESS (City, State, and ZIP Code) 5001 Eisenhower Avenue Alexandria, VA 22333-5600			
8a. NAME OF FUNDING/SPONSORING ORGANIZATION - -	8b. OFFICE SYMBOL (If applicable) - -	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER MDA903-85-K-0103			
8c. ADDRESS (City, State, and ZIP Code) - -		10. SOURCE OF FUNDING NUMBERS			

		PROGRAM ELEMENT NO. 6.11.02.A	PROJECT NO.2Q1611 02B74F	TASK NO. n/a	WORK UNIT ACCESSION NO. n/a

11. TITLE (Include Security Classification)

AUTOMATIC CONCEPT FORMATION IN A RICH INPUT DOMAIN

12. PERSONAL AUTHOR(S)
Michael Lebowitz

13a. TYPE OF REPORT Final Report	13b. TIME COVERED FROM Mar.85 TO Mar.87	14. DATE OF REPORT (Year, Month, Day) June 1988	15. PAGE COUNT 46

16. SUPPLEMENTARY NOTATION

Judith Orasanu, contracting officer's representative

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Artificial Intelligence Reasoning
			Computer Programs Learning (Computer)
			Expert Systems

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Learning by observation involves the creation of categories summarizing experience. In this research note, we summarize our research during the contact period with UNIMEM, an Artificial Intelligence system that learns by observation. UNIMEM is a robust program that can be run on many domains with real-world problem characteristics, such as uncertainty, incompleteness, and large numbers of examples. We give an overview of the program that illustrates UNIMEM's key elements, including the automatic creation of non-disjoint concept hierarchies that are evaluated over time. We then describe several experiments that we have carried out with UNIMEM, testing it on different domains (universities, Congressional voting records, and terrorist events), and an examination of the effect of varying UNIMEM's parameters on the resulting concept hierarchies. Finally, we discuss future directions for our work with the program.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT [X] UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL Judith Orasanu	22b. TELEPHONE (Include Area Code) 202/274-8722	22c. OFFICE SYMBOL PERI-BR

DD Form 1473, JUN 86   Previous editions are obsolete.   SECURITY CLASSIFICATION OF THIS PAGE

Table of Contents

ii

Automatic Concept Formation
in a Rich Input Domain

Michael Lebowitz
Department of Computer Science, Columbia University
New York, NY 10027

Executive Summary

Learning by observation involves automatic creation of categories that summarize experience. In this report we summarize our research during the contract period with UNIMEM, an artificial intelligence system that learns by observation. UNIMEM is a robust program that can be run on many domains with real-world problem characteristics such as uncertainty, incompleteness, and large numbers of examples. We give an overview of the program that illustrates UNIMEM's key elements, including the automatic creation of non-disjoint concept hierarchies that are evaluated over time. We then describe several experiments that we have carried out with UNIMEM, including testing it on different domains (universities, Congressional voting records, and terrorist events) and an examination of the effect of varying UNIMEM's parameters on the resulting concept hierarchies. Finally we discuss future directions for our work with the program.

1 Introduction

Learning from observation is a task that is important in domains where examples are not pre-classified, but where one still wishes to detect general rules and intelligently organize examples. In this paper we discuss UNIMEM, a system that learns from observation by noticing regularities among examples and organizing them into a generalization hierarchy. We view UNIMEM both as implementing an algorithm for concept formation and as a prototype intelligent information system that can incorporate large amounts of data into memory and retrieve appropriate information in response to user queries. UNIMEM is not intended to be a psychological model *per se*, since it deals with a task more data-intensive than people are likely to perform. However, in developing the program we have made use of techniques derived by observing human behavior.

The task of UNIMEM is to take a series of examples (or *instances*) that are expressed as collections of features and build up a generalization hierarchy of concepts. For example, UNIMEM might use information about a collection of universities to inductively determine the concepts of Ivy League universities, European technical universities, and so forth, and determine which examples are described by which concepts. The point of creating such concept descriptions is that they allow a performance element using the output of the program to make inferences about new examples based on partial information.

Successful learning from real-world input must deal with several constraints. The key features that characterize the operation of UNIMEM are:

- It learns *by observation*; it is not explicitly told how examples are grouped into concepts.
- It is *incremental*; after processing each example it must have available a generalization hierarchy; it cannot wait for all the input.
- It must handle *examples in large numbers* (currently hundreds, eventually more).
- Its generalizations are *pragmatic*; even when a generalization appears to apply, it does not have to; *usually* is good enough.[1]

Although certain learning systems have dealt with tasks having some of these characteristics, little work

---

[1] Pragmatic generalization is crucial in dealing with uncertain, incomplete or inconsistent data, where apparently equivalent situations may not always produce the same results.

has been concerned with all of them. However, all seem to characterize human concept formation and all seem valuable for learning in complex real-world domains. We constantly receive new examples and the world is not perfectly regular.

The task of UNIMEM is basically that of conceptual clustering as presented by Michalski and Stepp (1983), but our work also draws upon research in learning from examples (e.g., Winston, 1972, Mitchell, 1982, Dietterich and Michalski, 1986). However, in a learning by observation setting, one must consider not just how to compare examples, but also decide which examples to compare. This largely determines the concepts that one creates. We make the assumption that similarities among natural occurring examples reflect meaningful regularities in the world, an assumption that we discuss at length elsewhere (Lebowitz, 1986a).

The name UNIMEM is derived from the phrase UNIversal MEMory model, which reflects our goal of generality. We would like the system to be easily applicable to new domains, at least those where a feature-based representation is adequate. Domains that UNIMEM has been used on include: U. S. states, Congressional voting records, software evaluations, biological data, football plays, universities, terrorist events, census data, and financial data. In the following sections we provide an overview of the UNIMEM learning algorithm, along with an example of the system in operation, and then describe several experiments that we have performed with the program. These include both examining the system's behavior in several different domains and a study of the effects of varying UNIMEM's parameters. We conclude with with a discussion of several open research issues and the relation of our work to other research in machine learning.

2 The basic UNIMEM algorithm

UNIMEM takes a series of examples in a domain and organizes them into a permanent long-term memory.[2] The key idea behind UNIMEM is Generalization-Based Memory (GBM).[3] GBM is a hierarchy of

---

[2] UNIMEM runs on a DECSystem/2060 in UCI LISP and on an HP 9861 workstation and a VAX 750 running Portable Standard LISP.

[3] GBM is also used by our other prototype intelligent information system, RESEARCHER, which reads, remembers and generalizes from patent abstracts (Lebowitz, 1983a, 1986b). The instances in RESEARCHER are more complex than those in UNIMEM, but it can handle fewer examples. The idea of GBM was originally developed for IPP, a program that read and learned from news stories about terrorism (Lebowitz 1980, 1983b); see also Section 3.2 of this paper.

concepts describing classes of objects. GBM is built up by our systems, *including UNIMEM*, by generalizing specific examples. This involves both searching memory for similar examples and abstracting out similarities. To illustrate the UNIMEM learning algorithm, we will use examples from the domain of university information. For this domain we collected 284 universities descriptions of 224 distinct universities. Information was taken from standard reference books and by surveying undergraduate students. In studying learning by observation we feel that it is important to collect as much information as possible and not prejudge whether any particular piece of information is likely to be useful in generalization.

2.1 UNIMEM's representation of instances and concepts

Input to UNIMEM is a series of examples, or *instances*, given to the program one at a time. An instance is described as a set of *features* that are essentially attribute/value pairs.[4] Each university has attributes such as "percent of students receiving financial aid," "average math SAT score," and so forth. Some features, such as quality of social life, make use of arbitrary five point scales. While a simple feature representation is clearly inadequate for many tasks, it allows us to get started very easily on new domains. Table 1 shows the input features for Columbia, Yale, and Brown, three typical instances in the *university domain*.

The goal of UNIMEM is to recognize similar instances and abstract them to form a hierarchy of generalized concept descriptions. Instances are stored in GBM under the generalizations that describe them. The resulting concept hierarchy can, if desired, be used by a performance system, such as a question-answering program. The manner in which generalizations are related is illustrated in Figure 1, which shows part of a concept hierarchy formed by UNIMEM from 150 university instances averaging about 20 features apiece.[5] The complete hierarchy is shown in Appendix I. We can see in Figure 1 how

---

[4] UNIMEM actually uses attribute/facet/value triples. This greatly simplifies its use for frame-based representations. For example, in the terrorist event domain we use attributes and facets to distinguish among features of the different role fillers, e.g. the victim's nationality and the actor's nationality. However, for purposes of clarity, in this paper we have collapsed the attribute and facet fields.

[5] UNIMEM does not require that every instance have a value for every attribute, hence the number of features per instance varies. Also, attributes with multiple values are allowed.

Attribute	Value for COLUMBIA	Value for YALE	Value for BROWN
STATE	NEW-YORK	CONNECTICUT	RHODE-ISLAND
LOCATION	URBAN	SMALL-CITY	URBAN
CONTROL	PRIVATE	PRIVATE	PRIVATE
MALE:FEMALE	7:3	55:45	1:1
NO-OF-STUDENTS	< 5,000	< 5,000	< 5,000
STUDENT:FACULTY	9:1	5:1	11:1
SAT-VERBAL	625	675	625
SAT-MATH	650	675	650
EXPENSES	> $10,000	> $10,000	> $10,000
%-FINANCIAL-AID	60	40	40
NO-APPLICANTS	4,000-7,000	10,000-13,000	10,000-13,000
%-ADMITTANCE	30	20	20
%-ENROLLED	50	60	50
ACADEMICS	5 out of 5	5 out of 5	5 out of 5
SOCIAL	3 out of 5	3 out of 5	4 out of 5
QUALITY-OF-LIFE	3 out of 5	4 out of 5	5 out of 5
ACAD-EMPHASIS	LIB-ARTS	HISTORY	HISTORY
ACAD-EMPHASIS		BIOLOGY	BIOLOGY
ACAD-EMPHASIS		ENGLISH	ART-SCIENCES
ACAD-EMPHASIS		LIB-ARTS	

Table 1: Three instances of universities

the basic concept of a university is broken down into a number of more specialized versions. The hierarchical nature of the generalizations is indicated by indentation (e.g., GND60 inherits all the properties of GND2). The English concept descriptions have been added by hand. At the top level, we see universities under GND0 that are described by no generalized concepts. Shown beneath GND0 are two generalized concepts, GND2 and GND4. The later of these, which describes private universities, also has several more specific versions.[6]

In the hierarchy of generalizations that describe concepts of increasing specificity, instances and sub-generalizations are stored using efficient indexing methods.[7] The generalizations themselves are sets of features. Table 2 shows several of the generalizations taken from the hierarchy in Figure 1. GND4, the first generalization in the table can be summarizeo as "high-quality private universities", represented by an appropriate set of features. At this point in the run, no instances were stored directly under GND4, since those from which it was created had all been used to create sub-generalizations.

---

[6]The more specific versions of a generalization are referred to as its sub-generalizations.

[7]We have experimented with both discrimination networks (Feigenbaum, 1963; see Charniak et al., 1980 for implementation methods) and hash tables for indexing. The exact indexing method is not crucial in most domains; there are rarely a large number of instances under a given generalization, since sub-generalizations tend to be formed as the number of instances grows.

```
GND0 (``unusual'' universities that are not covered by any generalization)
     [DALLAS-BAPTIST-COLLEGE JUILLIARD MICHIGAN-STATE SUNY-BUFFALO
     UNIVERSITY-OF-MISSISSIPPI VASSAR]

  GND2 (high quality of life and academics; engineering emphasis)
       [CHALMERS-UNIVERSITY-OF-TECHNOLOGY ECOLE-POLYTECHNIQUE PENN-STATE
       SAN-JOSE-STATE UNIVERSITY-OF-CALIFORNIA-SAN-DIEGO UNIVERSITY-OF-TEXAS]

    GND60 (large state schools with strong social life
          [UNIVERSITY-OF-COLORADO UNIVERSITY-OF-MASSACHUSETTS-AMHERST]

    . . .


  GND4 (private universities with high academic level and medium social life)
       []

  GND9 (expensive, urban schools with strong applicant SAT scores)
       [HARVARD UNIVERSITY-OF-PENNSYLVANIA]

    GND119 (small schools with low admittance rates)
           [COLUMBIA WESLEYAN]

  GND19 (expensive schools with high enrollment yields)
       [MIT SWARTHMORE]

    GND133 (small schools with very high SATs and low admittance rates)
           [PRINCETON YALE]

    . . .


    . . .


  . . .

           Figure 1:  A portion of UNIMEM's concept hierarchy for a university run
```

As part of its representation, UNIMEM includes numeric ratings that indicate its confidence in each feature of each generalization. These numbers start at 0 and can increase or decrease during the processing of later examples, as described in Section 2.2.3. The values in the rightmost column of Table 2 are the confidence levels.[8]

The numbers in the third column of Table 2 are feature frequencies indicating how often each feature appears in other generalizations. This information is used for predictability analysis, a method for

---

[8] Naturally, the decimal places should not be taken too seriously. They are the product of the numeric evaluation procedure used.

ATTRIBUTE	VALUE	FEATURE FREQUENCY	FEATURE CONFIDENCE
GND4			
QUALITY-OF-LIFE	4 out of 5	1	20.00
ACADEMICS	4.5 out of 5	2	17.67
CONTROL	PRIVATE	3	16.00
SOCIAL	3 out of 5	3	20.00
[]			
GND9, a more specific version of GND4			
SAT-MATH	662.5	1	4.13
%-FINANCIAL-AID	60	1	5.00
LOCATION	URBAN	4	0.00
STUDENT:FACULTY	10:1	5	4.40
EXPENSES	> $10,000	5	11.00
[HARVARD UNIVERSITY-OF-PENNSYLVANIA]			
GND19, a more specific version of GND9			
SAT-VERBAL	637.5	1	2.72
%-FINANCIAL-AID	45.0	1	2.20
%-ENROLLED	55.0	2	1.00
NO-OF-STUDENTS	< 5,000	5	5.00
[MIT SWARTHMORE]			

Table 2: Selected concept descriptions for the university domain

determining which features are likely to indicate a generalization's relevance to new examples. While we will not discuss predictability in depth here -- it is discussed more fully in Lebowitz (1983b) -- the basic idea is that only certain features should be used to index a concept (because they indicate its relevance). and that these features can be identified efficiently using Generalization-Based Memory. Predictability analysis can also be important in determining causal explanations for generalizations (see Lebowitz. 1986c).

Table 2 also shows GND9, a more specific version of GND4, and GND19, a more specific version of GND9. The concept GND9 describes expensive, urban schools and GND19 describes schools that are, in addition, small with high verbal SAT scores. Each of these generalizations has instances (universities) stored with it. When future instances are found to be described by these generalizations, they will be compared to the examples stored there.

The use of a hierarchy of generalizations as a method of memory organization allows efficient storage of information since it supports inheritance. In addition, GBM allows the generalizations and

instances relevant for learning to be found efficiently in memory using the algorithm described below. This latter property is largely independent of UNIMEM's feature-based knowledge representation, as we have shown with RESEARCHER (Lebowitz, 1983a, 1986b) a system that uses a more complex representational scheme. The use of concept hierarchies with inheritance is by no means new; semantic networks (Quillian, 1968), frame systems (Minsky, 1975), MOPs (Schank, 1982) are among many formalisms that incorporate this approach. What distinguishes UNIMEM is the dynamic formation of the concept hierarchy and the use of this hierarchy to guide the development of further concepts.

An important part of the UNIMEM methodology is that the more specialized versions of a given concept need not be mutually exclusive. In Figure 1, for example, the two concepts "schools with high quality of life and academics; engineering emphasis" and "private universities with high academic level and medium social life" are obviously not mutually exclusive; a university could be described by both concepts. An implication of this is that UNIMEM can store an instance in several places in memory. Most clustering techniques require disjoint categories, but this does not seem to be the best way to maximize the inferential power of the concepts created. Even if a concept allows default inferencing, its negation may not, because the instances in that category have little in common. For example, universities that are neither in GND2 nor GND4 above may share no features, and hence no default inferences could be made based on identity in that class.

2.2 Adding new instances to memory

The basic process of incorporating a new instance into GBM makes direct use of the memory organization defined above. UNIMEM's incorporation algorithm for a new instance with a list of *input-features* can be broken into two phases:[9]

1. Search GBM for the most specific concept node(s) that the instance matches by calling SEARCH(*root-node, input features*).
2. Add the new instance to memory by calling UPDATE(*most-specific-node, input-features*) for the node(s) found by SEARCH. This involves comparing the new instance to the ones already stored and generalizing if appropriate.

---

[9]The UNIMEM incorporation algorithm includes a number of adjustable parameters, noted by a superscript P in the text. By parameterizing all aspects of UNIMEM, we do not give great meaning to any specific numeric value. In Section 4.1 we will discuss the possibility of setting the parameters automatically. Appendix II gives a complete listing of UNIMEM's parameters.

If desired the search phase could be used independently to retrieve instances that match an input description. This could be done for information retrieval and similar applications.

2.2.1 Searching the generalization hierarchy

As UNIMEM processes a new instance, it first finds the most specific generalizations that describe it. GBM can be viewed as a large discrimination net (Feigenbaum, 1963), so UNIMEM starts with its most general node and carries out a controlled depth-first search to find the most specific generalization(s) that legitimately describe the new instance. When the search begins, none of the input features have been matched to a generalization. As UNIMEM searches down the concept hierarchy, features are gradually accounted for by various generalizations. The major steps of the SEARCH algorithm are:

SEARCH(*node, unexplained-features*)
1. If the sum of the distances between the features in *unexplained-features* and those of *node* is "too large",[P] then *node* does not adequately match the instance; return the empty list.

2. Otherwise, for each potentially relevant sub-node *sx* of *node*, call SEARCH(*sx,* [*unexplained-features* - features of *node*]).

3. If for any *sx,* SEARCH returns a list of nodes that describe the new instance, then return the union of those lists.

4. Otherwise, return the singleton list of *node*. (This case occurs only when each sub-node conflicts with the new instance. Since *node* does not conflict with the new instance, it is the most specific acceptable generalization on this search path.)

During UNIMEM's search process, feature values can do more than match or mismatch – there can be varying degrees of closeness.[10] Instead of values simply matching or not, we allow the quality of feature matches to vary between 0 (total mismatch) and 1 (perfect match).[11] When UNIMEM matches a new instance to a generalization, it considers whether the sum of the distances between the features in the generalization and those in the new instance is small enough[P] to assume that the generalization describes the instance.[12]

If an instance has feature values that conflict with a generalization, which is allowed as long as the total conflict is not too high, then the instance feature values simply override those in the generalization.

---

[10]We developed categorization algorithms for numeric input that allowed an all-or-none regimen to work reasonably well (Lebowitz, 1985), but we have since modified UNIMEM to take into account the closeness of values as described here.

[11]The system is set up so that a user can easily define different distance measures for various features, if desired. We currently consider numeric data, ordinal data, and simple hierarchical data.

[12]We also add in a penalty for any feature of the generalization simply missing from the instance. This is possible since instance descriptions can be incomplete.

This contrasts with many learning techniques which assume that all the features of a generalization must hold for each instance that it describes. In early experiments with UNIMEM, we found that such an all-or-none matching scheme led to the creation of excessive numbers of slightly different generalizations because new instances did not quite fit under old ones. Allowing contradiction does potentially leave UNIMEM open to problems of the sort described by Brachman (1985), such as describing an instance as "an Ivy-League type school except it's not in the East, not private, not expensive ...". However, as long as we keep the allowed-difference parameter small, this does not appear to happen.

2.2.2 Storing a new instance in memory

Once UNIMEM has retrieved the most specific generalization(s) that a new instance matches, it compares the instance against others already stored with the concept(s) to determine whether further generalizations should be made. The system looks for instances that have features in common with the new one. If it finds one that has enough features in common,[P] it creates a new node by generalizing the common features, and it stores the contributing instances with the new concept. If no sufficiently similar instances are found, it stores the new instance under the existing generalization. The attribute/value representation of UNIMEM normally yields a unique generalization of two instances[13] but multiple generalizations are sometimes created by matching a new instance with several existing ones. The main steps in the UPDATE algorithm are:

UPDATE(node, new-instance)
    1. Define new-features as the features of new-instance that are not part of node (or its parent nodes). This information is retained from SEARCH.

    2. If none of the instances currently stored under node have enough[P] features with values sufficiently close[P] to those of new-instance to warrant a new generalization, then store new-instance under node.[14]

    3. Otherwise, for each instance with enough features in common with new-instance, create a generalization node comprised of the shared features and:

        a. Store the new node in the node's sub-generalization list.

        b. Store both instances under the new node.

        c. Remove the old instance from the original node.

---

[13]Exceptions would be if there are multi-valued attributes or if the "averaging" process described below returns multiple possibilities.

[14]"Enough" is defined as a percentage of the maximum number of features of the two instances being compared.

In deciding which features to include in a generalization, UNIMEM selects all those in the two instances with values that are sufficiently close[P]. In those cases where features have slightly different values, UNIMEM uses an "average" feature value in the generalization. For real-values features this is the arithmetic or geometric mean; for ordinal attributes it is one of the two values; and for hierarchically-ordered attributes it is the lowest common ancestor.

2.2.3 Evaluating generalizations

As seen above, concepts are generalized by UNIMEM on the basis of only two instances. This can cause the creation of an over-specified generalization if the initial instances share spurious features. Generalizations can be under-specified if the instances had unknown values for relevant features (which is possible, since UNIMEM does not require every instance to have values for each feature). Under-specification is not a problem, since the missing features will simply appear in sub-generalizations. However, concepts must be evaluated when they are potentially relevant to future input in order to remove overly-specific features. This is particularly true in domains where there are a large number of features for each instance, since coincidental matches become inevitable. UNIMEM performs evaluation as a normal part of the memory search process, since the generalizations to be evaluated are exactly those that are accessed when a new instance is processed. We simply add the following step to the beginning of the SEARCH algorithm:

- Increase confidence in any features of *node* that is also in *unexplained-features*; decrease the confidence of those that are contradicted.[15] Delete any features with confidence levels that go low enough[P]. Make permanent any features with confidence levels that go high enough[P] (e.g., stop modifying their confidence levels).

The evaluation operations are applied to all nodes considered during the SEARCH process, even if they do not ultimately match.

This modification to SEARCH does not lead UNIMEM to entirely eliminate a generalization when it fails to fit later input. Instead, it tries to throw away just the "bad" (overly specific) parts and keep the "good" parts. Confidence modification occurs by incrementing confidence levels when new values are close[P] to the generalization (in terms of the distance measure) and decrementing them when they are

---

[15]*Node* is guaranteed to be a "potentially relevant" node by the way the algorithm is structured.

11

not. The amounts of the increments or decrements depend upon the distance between the feature values of the instance and of the generalization. If a confidence level falls below a negative threshold,[P] then the system eliminates that feature from the generalization, since it has unreliably appeared in instances when the generalization seemed relevant.[16] Above a specified level[P] values are "frozen" and assumed to be permanently correct.

In some cases the feature evaluation process leads to concepts so general that they no longer provide substantial information. There is not advantage to retaining a category with so few features that no inferences can be made when an instance is matched to it. Thus, UNIMEM eliminates an entire generalization when too few of its features[P] remain, defined as a percentage of the number of features of the instances that formed the generalization. When it deletes a generalization, UNIMEM also loses access to the instances and sub-generalizations stored there. This will lose instances that are not also stored elsewhere, but if we immediately reindex the instances with the parent node, then the same instances that initially formed the eliminated generalization will do so again. In the domains that we deal with there are enough input examples so that good concepts will eventually be created, despite losing some information. However, for other domains different strategies might be appropriate, such as putting deleted instances back into memory after a delay.

2.3 A simple program trace

To illustrate UNIMEM's update algorithm, we will look in detail at how it adds an instance from the university domain to an existing memory.[17] This example will involve the three universities described in Table 1 as well as six others with descriptions that can be found in Appendix III. Figure 2 shows the structure of UNIMEM's memory after the instances MIT, Brown, Princeton, Harvard, Yale, Arizona State, Case Western, and Auburn have been processed in that order. Table 3 shows the details of the generalizations in this sample run.

---

[16]Other than removing features, UNIMEM does not use the confidence level in the matching process. An interesting extension might be to give added weight to features with high confidence values.

[17]To make pedagogic points, we have set some of UNIMEM's parameters to unrealistic values.

```
                        ------
                        |root|
                        |    |
                        ------
              /            |            \
        ------          ------          ------
        |GND1|          |GND2|          |GND4|
        |1.31|          |1.24|          |2.68|
        ------          ------          ------
        Brown (1)       Harvard (1)     Case Western
        MIT (4)            |               |
                        ------          ------
                        |GND3|          |GND5|
                        |1.98|          |    |
                        ------          ------
                        Princeton       Arizona State
                        Yale            Auburn
```
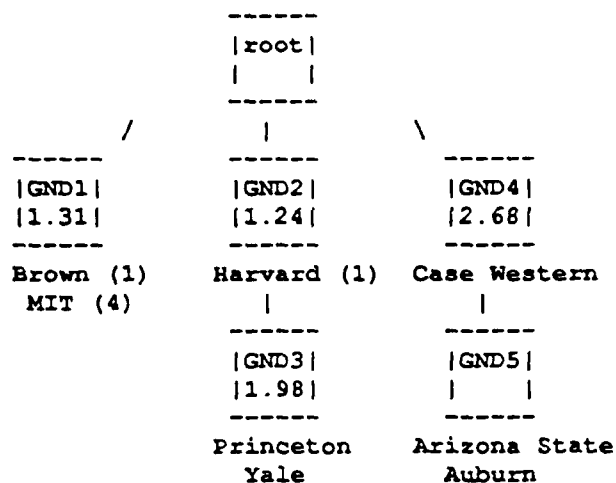
Values with GND1, GND2, GND3, and GND4 are the sum of the feature
distances between that node and Columbia. The numbers in parentheses
are the number of features the instance has in common with Columbia,
not including the ones accounted for by generalizations.

Figure 2: Initial memory structure for the sample run

We will describe in some detail how UNIMEM now processes Columbia. It begins by searching memory for the most specific generalizations that satisfactorily match the new instance. This begins by matching Columbia's features with those of GND1. As shown in Figure 2, the total difference between the features of GND1 and Columbia is 1.31. As the parameters were set for this run, the allowed difference is 1.36 (8% of 17 features), so GND1 is considered acceptable. The same is true for GND2, with its 1.24 difference. However GND2's sub-generalization, GND3 is not acceptable, nor is GND4. Note that since GND4 is not acceptable, its sub-generalization, GND5 was not even considered. The final result of the search is that GND1 and GND2 are the most-specific generalizations that match Columbia.

Searching through memory also involves the updating of feature confidence levels. In this case, as a match is conducted, if a generalization features value is close to that of Columbia, then its confidence level is increased, otherwise it is decreased. The amount of the increment or decrement is based upon the degree of the match or mismatch. Looking at GND1, for instance, if we compare the rightmost two columns of Table 3, we can see that the confidence level for the percentage of financial aid went down, since the generalization value was 45% compared to 60% for Columbia. The remaining confidence levels went up, as the Columbia values were quite close to the values in GND1. The confidence levels for GND5 were not adjusted at all, as it was skipped by the search algorithm.

ATTRIBUTE	VALUE	FEATURE FREQUENCY	INITIAL FEATURE CONFIDENCE	FINAL FEATURE CONFIDENCE
GND1				
STUDENT:FACULTY	5:1	1	5.78	6.75
SAT-VERBAL	637.5	1	-2.00	-1.25
%-FINANCIAL-AID	45.0	1	1.60	1.40
%-ADMITTANCE	25.0	1	-1.20	-0.60
%-ENROLLED	55.0	1	0.80	1.40
SOCIAL	3.5 out of 5	1	3.33	4.00
NO-OF-STUDENTS	< 5,000	1	0.00	1.00
LOCATION	URBAN	1	-1.00	0.00
EXPENSES	> $10,000	2	2.00	3.00
ACADEMICS	5 out of 5	2	1.33	2.33
CONTROL	PRIVATE	2	2.00	3.00
GND2				
%-FINANCIAL-AID	55.0	1	1.20	1.80
%-ADMITTANCE	20	1	-2.00	-1.80
SOCIAL	3 out of 5	1	2.00	3.00
QUALITY-OF-LIFE	3.5 out of 5	1	2.00	2.67
ACAD-EMPHASIS	HISTORY	1	-2.00	deleted
ACAD-EMPHASIS	LIBERAL-ARTS	1	-2.00	-1.00
MALE:FEMALE	65:35	1	-0.68	-0.12
STUDENT:FACULTY	7:1	1	3.89	4.88
SAT-MATH	675	1	-0.50	0.00
EXPENSES	> $10,000	2	0.00	1.00
ACADEMICS	5 out of 5	2	-0.67	0.33
CONTROL	PRIVATE	2	0.00	1.00
GND3, a more specific version of GND2				
SAT-VERBAL	662.5	1	0.00	0.25
%-FINANCIAL-AID	45.0	1	0.00	0.00
NO-APPLICANTS	10,000-13,000	1	0.00	-1.00
%-ENROLLED	60	1	0.00	0.20
NO-OF-STUDENTS	< 5,000	1	0.00	1.00
ACAD-EMPHASIS	HISTORY	1	--	0.00
GND4				
STUDENT:FACULTY	20:1	1	1.00	1.97
%-ADMITTANCE	82.5	1	0.40	-0.60
ACADEMICS	3 out of 5	1	0.33	0.00
ACAD-EMPHASIS	ENGINEERING	1	1.00	0.00
GND5, a more specific version of GND4				
CONTROL	STATE	1	0.00	0.00
MALE:FEMALE	50:50	1	0.00	0.00
SAT-VERBAL	465.0	1	0.00	0.00
SAT-MATH	522.5	1	0.00	0.00
%-FINANCIAL-AID	50	1	0.00	0.00
%-ENROLLED	60	1	0.00	0.00
SOCIAL	4	1	0.00	0.00
QUALITY-OF-LIFE	4.5	1	0.00	0.00
GND6, a more specific version of GND1				
MALE:FEMALE	75:25	1	--	0.00
%-FINANCIAL-AID	55.0	1	--	0.00
NO-APPLICANTS	4,000-7,000	1	--	0.00
QUALITY-OF-LIFE	3 out of 5	1	--	0.00

Table 3: Generalizations of the sample run

While searching GND2, the confidence level for history as an academic emphasis was reduced from -2.0 to -3.0. This caused the confidence level to go below the threshold for retaining features, so the feature was deleted from the generalization. In order to maintain correctness, the same feature had to be added to GND2's sub-generalization, GND3. Also, since the feature was deleted from GND2 during the matching process, this particular feature difference was not considered part of the total discrepancy between GND2 and the new instance, which allowed a match with Columbia.

With the search and confidence evaluation phase complete, UNIMEM updates memory by adding Columbia to both GND1 and GND2. In each case, it compares the new instance to those already stored with the generalization to see if there are a significant number of features in common (other than those already accounted for by the generalization). The number of features that Columbia has in common with each relevant instance is shown in parentheses in Figure 2. *Columbia only shares one feature with* Brown, the first instance under GND1, but four with MIT. Since this is above the parameter for generalizing on this run, UNIMEM creates a new generalization, GND6, which is indexed under GND1. Both MIT and Columbia are stored under the new generalization. Harvard, the only instance under GND2, the other generalization that Columbia matched, shares only one feature with the new instance, and so no generalization is made. Columbia is simply stored under GND2. The structure of memory at the end of the processing of Columbia is shown in Figure 3.

```
                              ------
                             |root |
                             |     |
                              ------
                  /             |             \
     ------             ------             ------
    |GND1 |            |GND2 |            |GND4 |
    |     |            |     |            |     |
     ------             ------             ------
    Brown              Harvard            Case Western
      |                Columbia              |
      |     '            |               ,  |
     ------             ------             ------
    |GND6 |            |GND3 |            |GND5 |
    |     |            |     |            |     |
     ------             ------             ------
    Columbia           Princeton         Arizona State
    MIT                Yale              Auburn
```
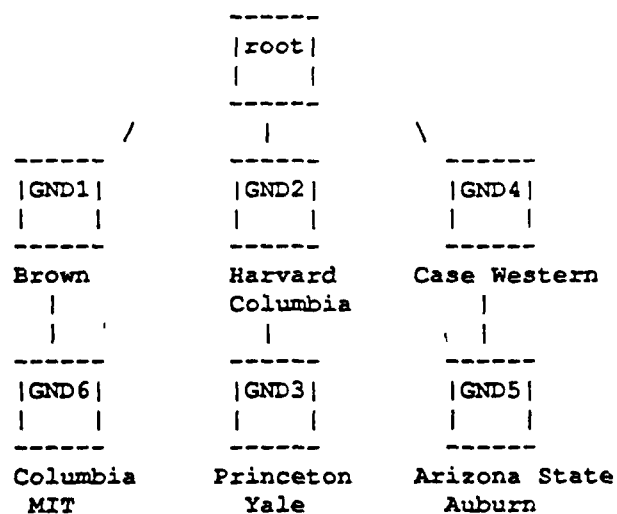
Figure 3: Final memory structure

Notice that Columbia is not compared against any of the instances under generalizations other than GND1 and GND2. It is possible that it has much in common with some of these instances. However, it is much more likely to be similar to those under the matched generalizations. Restricting the set of instance that we match against is a prime factor in maintaining the efficiency of the algorithm.

This sample UNIMEM run also illustrates the nature of the disjoint UNIMEM generalizations. GND1 and GND2 are not mutually exclusive, and the program has matched Columbia with both of them. Essentially, GND1 covers small urban universities with high academic levels and GND2 covers high "quality of life," liberal arts schools. Columbia can quite logically be considered to exemplify either concept.

2.4 UNIMEM in terms of search and memory organization

Like artificial intelligence programs in general, UNIMEM can be viewed as searching through a space of alternatives. In this case, each state in the space represents an entire concept hierarchy. UNIMEM employs several operators to move through this search space, all of which are driven by the addition of new instances. First, it can simply change the confidence levels of features in concepts that appear relevant to a new instance. Second, it can modify concepts by removing features for which the confidence levels fall too low. Third, it can modify the structure of the generalization hierarchy by adding new concepts when instances are sufficiently similar. Finally, it can delete generalizations (and all their sub-generalizations) when too few features remain after deletions.

Although it is possible to describe UNIMEM in search terms, we feel it is more valuable to describe it in the memory terms that we have been using. The basic data structure of Generalization-Based Memory is the key to its operation. In fact, we feel that more researchers should consider their work in memory terms. Viewing learning from this perspective forces one to consider how the concepts that are created can be efficiently accessed, how memory should be modified, and how the various data structures evolve over time, both in terms of structure and size.

16

3 Experiments with UNIMEM

An important criterion on which to evaluate any learning system is its generality. In this section we demonstrate UNIMEM's behavior in two additional domains: congressional voting records and terrorist events. Another important issue concerning a system's performance is how it responds to changes in parameter values. Thus we conducted a set of experiments in parameter variation which we also present in this section.

3.1 Congressional voting records

One domain on which we tested UNIMEM involved Congressional voting records. Instances were formed from the votes of each U.S. Congressman on a number of major issues (taken from *The 1983 American Political Almanac*) combined with information about the district and state represented. One advantage of this domain for research purposes is that people have strong intuitions about the kinds of generalizations that should be found. A complete description of the domain can be found in Lebowitz (1986c). In the run described here, we presented UNIMEM with 100 instances, each containing 15 votes and about 21 other features.[18] We expected to find generalizations that related the various votes to each other (e.g., "liberal" and "conservative" ideologies), along with others that related the votes to the states and districts represented (e.g., someone representing a highly urban state would support bills that help cities). Indeed UNIMEM formed concepts of this sort.

Table 4 shows several of the resulting generalizations from this run and Figure 4 shows their organization in memory. One top-level generalization, GND2, describes congressmen from agricultural states with high levels of school expenditures[19] who voted for an education bill, parks in Alaska, and so forth. The 24th Texas Congressional District is stored under this generalization, along with two sub-generalizations. Someone familiar with U.S. politics would describing this votingpattern as "liberal". Similarly, the second top-level node in this example, GND3, would be considered "conservative".

---

[18]Not all instances had all features.

[19]Values of the form "n out of m" represent categorized numeric information. In this domain categories were automatically created using methods described in Lebowitz (1985).

```
                          ------
                         | root |
                          ------
                       /          \
              ``liberal''          ``conservative''
                ------               ------
               | GND2 |             | GND3 |   . . . .
                ------               ------
               /      \                 |
         ------  ------             ------
        | GND4 | | GND7 |  . . .    | GND8 |   . . .
         ------  ------             ------
```
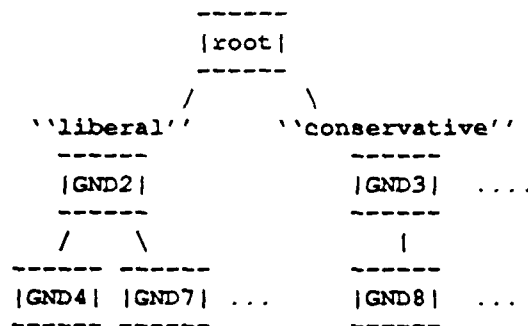
Figure 4: *CD generalization structure*

These two generalizations are non-disjoint, since their features do not include opposite votes on the same bills. Instead, the generalizations include votes on different bills and are not exclusive. A conservative record can most confidently be identified based on the votes shown in GND3, such as a vote against cutting the MX missile, while a liberal record shows up from the votes in GND2, such as a positive education vote. A Congressman can fit into both categories (indeed, we see this happen in the sub-generalizations of GND2). Apparently "liberal" does not equal "not conservative".

The situation becomes particularly interesting when we look at the sub-generalizations of GND2 (GND4 and GND7) and GND3 (GND8). When we examine these generalizations carefully, we see that the contrasting votes omitted from the top-level generalizations appear in their sub-generalizations. For example, the "liberal" generalization (GND2) contains a vote against a cut in social funds. The converse of this vote does not appear in GND3, but it is present in its sub-generalization, GND8. Similarly, the opposite of the conservative vote against the MX missile is not included in GND2, but it does occur in one of the sub-generalizations, GND7. Certain votes that do not serve well to define concepts at the top level can be useful in refining these concepts after the initial set of features is "factored out."

3.2 Terrorist events

UNIMEM was developed from the memory and generalization module of IPP (Lebowitz, 1980, 1983b, 1983c), a program that read news stories about international terrorism and added them to long-term memory. In the process, it formed a generalization hierarchy using a learning module that we will

```
ATTRIBUTE                      VALUE

GND2
STATE-INDUSTRY                 AGRICULTURE
STATE-SCHOOL-EXP               3 out of 3
EDUCATION-VOTE                 FOR
ALASKA-PARKS-VOTE              FOR
SOC-FUND-CUT-VOTE              AGAINST
STATE-INDUSTRY                 MANUFACTURING
STATE-INCOME                   3 out of 4
STATE-MINORITY-PCT             1 out of 2
[TEXAS24]


GND4, a more specific version of GND2
WIND-TAX-LIM-VOTE              AGAINST
GAS-CONT-BAN-VOTE              FOR
HOSP-COST-CONT-VOTE            FOR
NICARAGUA-BAN-VOTE             AGAINST
PARTY                          DEMOCRATIC
STATE-FARM-VAL                 5 out of 6
OSHA-CUT-VOTE                  AGAINST
FOOD-STAMP-CAP-VOTE            AGAINST
PAC-LIMIT-VOTE                 FOR
STATE-URBAN-PCT                6 out of 6
FAIR-HOUSING-VOTE              FOR
[CALIFORNIA6 CALIFORNIA7 CALIFORNIA8 ...]


GND7, a more specific version of GND2
NICARAGUA-BAN-VOTE             AGAINST
STATE-DEBT                     5 out 7
STATE-INDUSTRY                 TOURISM
MX-CUT-VOTE                    FOR
DISTRICT-POP-DIR               UP
STATE-FARM-VAL                 5 out of 6
FAIR-HOUSING-VOTE              FOR
[FLORIDA13 FLORIDA15 MICHIGAN10 ...]


GND3
HOSP-COST-CONT-VOTE            AGAINST
WIND-TAX-LIM-VOTE              FOR
DRAFT-VOTE                     FOR
NUC-POWER-VOTE                 AGAINST
MX-CUT-VOTE                    AGAINST
STATE-TAXES-PERCAP             2 out of 5
DISTRICT-POP-DIR               UP
STATE-INCOME                   3 out of 4
STATE-INDUSTRY                 MANUFACTURING
STATE-MINORITY-PCT             1 out of 2
[]

GND8, a more specific version of GND3
STATE-POPULATION               6 out of 7
EDUCATION-VOTE                 AGAINST
PARTY                          REPUBLICAN
NICARAGUA-BAN-VOTE             FOR
GAS-CONT-BAN-VOTE              AGAINST
SOC-FUND-CUT-VOTE              FOR
OSHA-CUT-VOTE                  FOR
STATE-URBAN-PCT                6 out of 6
PAC-LIMIT-VOTE                 AGAINST
[CALIFORNIA34 FLORIDA10 ...]
```

Table 4: Congressional voting record domain generalizations

refer to as IPP-MEM. An interesting aspect of this domain is that descriptions of events tend to be

19

incomplete, so that the instances do not have the same feature sets. UNIMEM differs from IPP-MEM in a number of technical ways. For example, parameters have been added to make it more flexible and different methods of low-level indexing are available. The most substantial change is the modification of confidence methods to consider each feature in a generalization separately. IPP-MEM maintained a single confidence level for each generalization. As a result, even one anomalous feature could cause an entire generalization to be deleted. We wanted to see whether this change in UNIMEM would dramatically alter the kinds of generalizations that remain in the generalization hierarchy at the end of a run.

The experiment described here used 374 of the stories that the IPP text processor (IPP-NLP) handled most accurately, all taken from the period of 1979-1980.[20] Table 5 shows three successively more specific generalizations that UNIMEM built up from a number of bombing stories in the sample set. The features in Table 5 with "deleted" in their confidence fields have been removed and are not part of the final generalizations (but were initially included). GND10 describes terrorist events involving bombs in Western Europe in which people were hurt. This generalization was originally formed from stories originating in Spain with an explosion taking place. While this made the generalization carry more information than the final version, it was also less widely applicable. Since other stories were found with the same characteristics, but not occurring in Spain, UNIMEM removed the location from the generalization. This allowed it to apply to a wider range of examples. Ultimately UNIMEM created a sub-generalization of GND10 (GND30) that described events in which an explosion took place and people were killed (as indicated by the -10 health value). The system also formed an even more specific variant of GND10 (GND37) in which the victims were soldiers.

The output in Table 5 is quite typical of the performance of UNIMEM in the terrorist event domain. It created concepts that seemed to capture basic regularities in the domain. Qualitative comparison of the UNIMEM generalizations in the terrorist event domain with those generated by IPP-MEM was quite

---

[20]The processing needed to prepare IPP-NLP output for UNIMEM is described in Appendix IV.

ATTRIBUTE	VALUE	FEATURE FREQUENCY	FEATURE CONFIDENCE
GND10			
WEAPON-WEAPON	BOMB	1	6.50
WEAPON-CLASS	EXPLOSIVE	1	15.00
RESULTS	HURT-PERSON	2	15.00
LOCATION-AREA	WESTERN-EUROPE	2	15.00
METHODS	$EXPLODE-BOMB		deleted
S-MOP	S-DESTRUCTIVE-ATTACK		deleted
VICTIM-NATIONALITY	*SPAIN*		deleted
LOCATION-NATION	*SPAIN*		deleted
[S214 S253B S375A]			
GND30, a more specific version of GND10			
RESULTS-HEALTH	-10	1	15.00
S-MOP	S-DESTRUCTIVE-ATTACK	4	11.00
METHODS	$EXPLODE-BOMB	4	9.75
VICTIM-NATIONALITY	*SPAIN*		deleted
[S89A S138 S139 S185 S220 S222A S260 S263 S321A S334 S337 S534D]			
GND37, a more specific version of GND30			
VICTIM-ROLE	AUTHORITY	1	3.75
VICTIM-ROLE	SOLDIER	1	-0.75
VICTIM-AUTH	T	1	3.75
VICTIM-POL-POS	ESTAB	1	3.75
VICTIM-NATIONALITY	*ENGLAND*		deleted
LOCATION-NATION	*N-IRELAND*		deleted
[S29 S223A S241 S314]			

Table 5: An IPP-NLP/UNIMEM bombing example

informative. Overall, the UNIMEM generalizations seemed more intuitively plausible and covered a wider range of concepts. On the other hand, they also seemed more "bland", omitting some of the most "interesting" generalizations that the original system had made -- that pistols with silencers were frequently used in shootings of Italian political activists, for example.

It is clear from Table 5 why the UNIMEM generalizations were more "bland" than those of IPP. Suppose that each system formed a complicated generalization, like the one above, by noticing similar events. In response to future data, IPP-MEM would either keep the description *in toto* or delete it entirely. On the other hand, UNIMEM would inevitably refine the generalization, and make it less unusual, by removing the coincidental elements so that it covers a wider range of events. While this is mildly disappointing in the short run, overall it is quite positive. UNIMEM produces the basic generalizations (e.g., terrorist shootings usually hurt people) needed for default reasoning. Furthermore, the "flashy" generalizations need not be lost, as they can be formed as sub-generalizations. This did not happen very

often in our experiment with the terrorist domain, since there were not enough examples and, more importantly, many of the examples had very few features. Large numbers of features actually hindered IPP-MEM, as it had no way to refine over-generalized concepts. Given UNIMEM's ability to deal with greater numbers of features, we plan to increase the level of detail of the feature sets produced from IPP-NLP representations.

3.3 The effect of varying UNIMEM parameters

UNIMEM has a number of adjustable parameters that affect its behavior. Given different parameter settings, the same sequence of instances can lead to many different generalization hierarchies. In order to generate the "best" hierarchy, we will have to find appropriate parameter settings, which may vary among domains or applications. For example, one might aim for generalizations that predict a great deal in a limited number of situations, or for ones that are widely applicable but predict only a small amount of information.[21] Convergence rate is also an issue for an incremental system like UNIMEM. Depending on the degree of consistency in the domain in question, one may have to trade off learning speed with various aspects of hierarchy quality.

In order to better understand the effect of UNIMEM's parameters on the shape of the hierarchy created, and the convergence behavior, we conducted a series of experiments involving parameter variation, which are described in this section.

3.3.1 Evaluating UNIMEM behavior

In order to intelligently evaluate the output of UNIMEM, we must consider what makes one set of concepts better than another. We can apply the criteria recursively so that they applies to entire hierarchies. Other things being equal, we would prefer concept descriptions with many features, since each additional feature adds inferential power to the generalization. However, the more specific a generalization, the fewer examples it can be expected to cover. Thus, there is an inherent trade-off in concept formation between coverage and the ability to make predictions based on the generalizations.

---

[21] Gluck and Corter (1985) and Fisher (1986) have argued on information-theoretic grounds that there is an optimal level of classification. However, their work does not apply directly to non-disjoint categories, nor to situations in which the input is uncertain and incomplete.

A second trade-off in concept formation involves non-disjoint concepts. As pointed out earlier, allowing overlap will often result in more specific generalizations with more inferential power. However, overlap can also make the concepts less useful for a performance element, as it will have to consider how to deal with the case where a new example fits into several categories. In addition, if there are two concepts that are only slightly different, since many of the same instances will be stored under both, UNIMEM will create very similar trees of sub-generalizations, which is inefficient in both space and time.

The trade-offs between concept specificity and both generality and minimal overlap can be instantiated in UNIMEM terms with two criteria. First, under any given generalization, there should be a "modest number" of sub-generalizations. A number in the 4-12 range seems appropriate in our domains as it yields generalizations that are relatively specific, but general enough to cover a range of instances. Second, the instances covered by a set of concepts should be divided roughly equally among them, guaranteeing that each generalization describes a number of different instances and tending to minimize overlap.

Since UNIMEM forms concepts incrementally, we must also deal with convergence. It is important to look at how long it takes the program to settle on a set of high confidence concepts that it is not likely to invalidate later in the run. Although we would like the generalization hierarchy to converge as rapidly as possible, as we will see below, this goal may conflict with the other desired properties. However, we must make sure that the program does not simply continually create and invalidate concepts.

3.3.2 An experiment in parameter variation

Our initial experiment involved the variation of two parameters -- the percentage of features that instances must have in common for generalization to occur and the percentage of features that must remain in a generalization for it to be retained.[22] Specifically, we set the "percentage to generalize" parameter value at 25%, 37.5%, and 50% and the "percentage to retain a generalization" at 20% and

---

[22]The percentage to retain a generalization parameter is computed in terms of the initial number of features in the instances.

25%.[23] We expected these parameters to influence UNiMEM's rate of generalization, e.g., the larger percentage of features required for a generalization, the slower the system should generalize.

The experiment involved three randomly selected sequences of 100 universities apiece. For each of the six pairs of parameter values, we had UNIMEM independently incorporate the three sets of universities into an initially empty memory and then collected summary information. All of the data were averaged across the three runs.[24] Given difficulties in making assumptions about the distribution of the data, we will not present statistical analyses, but instead examine the data qualitatively. In addition, we restricted our analysis to the top-level generalizations, which can be viewed as UNIMEM's overlapping categorization of all the input instances.

The first dependent variable that we measured was the number of top-level generalizations retained by UNIMEM, as displayed in Figure 5. In the various experimental conditions the system retained an average of between 9 and 14 such generalizations, although the number will inevitably approach zero if either parameter is made very much higher. There is some indication that the number of remaining generalizations tends to increase along with each parameter, but this is not a strong trend.

In an attempt to clarify these results, we examined the two dependent variables that determine the number of generalizations that remain -- the number that are created and the percentage of those created that are deleted. The average number of generalizations created for each combination of parameters is shown in Figure 6. We see, somewhat surprisingly, that the number of generalizations created declines only moderately as the features needed to generalize increases. We might expect this decline to be greater since it should be harder to find instances with more features in common. For reasons that will be considered below, the number of features needed to *retain* a generalization has a substantial effect on the number *created*.

---

[23]Instances contained about 20 features in this domain, so the absolute number of features needed to retain a generalization is roughly 4 at the 20% level and 5 at the 25% level. The 25% value for the features to generalize parameter requires about 5 features, the 37.5% value requires about 8, and the 50% value about 10.

[24]While UNIMEM is potentially susceptible to effects of the order of instances, this usually is not a major issue. A few odd generalizations made at the beginning of a run may have to be discarded, losing some information. In this experiment, while there was some variation in the results between the three different data sets, in no case was it striking.

```
Average number of generalizations at end of run
14                          k              K
13                        ╱ 14            14
12                      ╱ K
11                    ╱ 12              ╲  k
10                  ╱                      11
 9      kK       ╱
 8       9
 7
 6
 5
 4
 3
 2    K -- pct to retain gen = 25%
 1    k -- pct to retain gen = 20%
 0
    -------------------------------------------------
         25%          37.5%          50%
         Feature percent to generalize
```
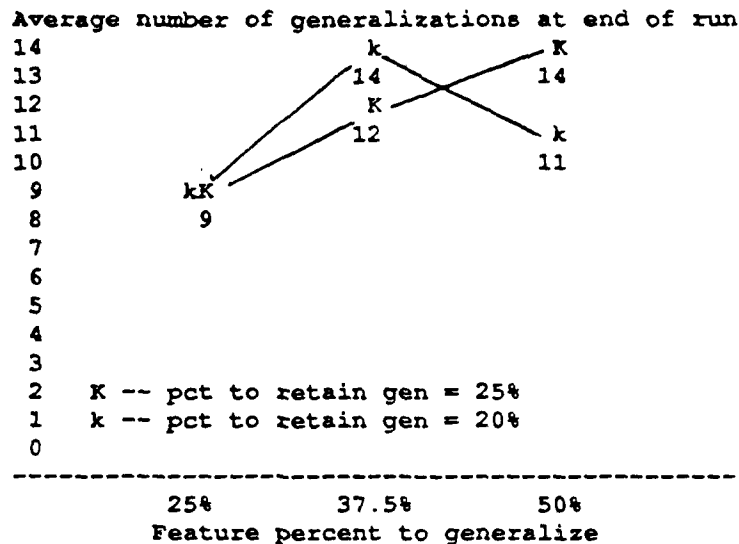
Figure 5: Generalizations at end of run as a function of percentage to generalize and percentage to retain

```
Average generalizations created
36
33        C ─────── C ─────
30        32          32        ╲  C
27                                 29
24        c
21        24  ╲     c
18             ╲   21  ───────
15                              c
12                              17
 9
 6    C -- pct to retain gen = 25%
 3    c -- pct to retain gen = 20%
 0
    -------------------------------------------------
         25%          37.5%          50%
         Feature percent to generalize
```
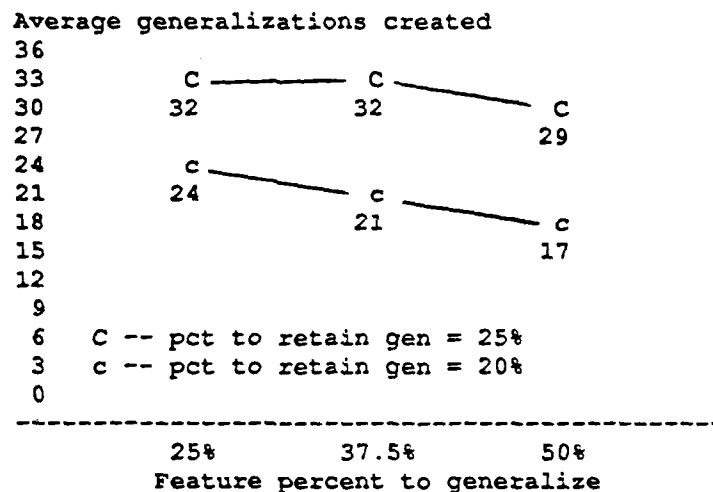
Figure 6: Generalization creation as a function of percentage to generalize and percentage to retain

Figure 7 shows the average percentage of generalizations deleted by UNIMEM's evaluation method when too many features were removed. As expected, *more generalizations are deleted at the 25% retention level than at the 20% level.* A more surprising result is that number of features needed to create a *generalization affects the percentage that are deleted.* The reason becomes clear when one realizes that the more features that are initially in a generalization, the more that can be removed and still

25

be over the deletion threshold. In effect, requiring more common features to form a generalization enhances the possibility that there will be a "good" set of features included that UNIMEM can retain once the "bad" ones are whittled away.

```
Average percentage of generalizations deleted
90
80        71%
70        D
60        d              D
50        63%           63%            D
40                                    51%
30                      d──────────── d
20                     33%           35%
10   D -- pct to retain gen = 25%
 0   d -- pct to retain gen = 20%
    ─────────────────────────────────────────────
        25%          37.5%          50%
        Feature percent to generalize
```
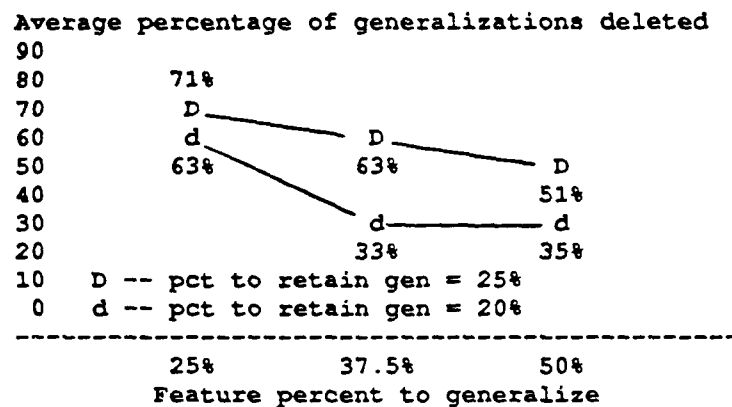
Figure 7: Generalization deletion as a function of
percentage to generalize and percentage to retain

The decrease in the deletion rate as one increases the percentage of features needed to generalize explains the decrease in the creation rate. Since fewer generalizations are deleted, there is a higher chance that new instances will be stored under existing generalizations before the hierarchy converges. This diminishes the chance that new top-level generalizations will be created. The combination of creation and deletion behavior provides an explanation for the smaller number of generalizations retained at both ends of the 20% deletion level curve in Figure 5. If the number of features needed to generalize is very low, then few generalizations are kept, and if it is very high, then few are made. Determining the robustness of this phenomenon will require the collection of further data.

Another evaluation criteria that one might expect the parameters under consideration to influence is the average number of features in a generalization. Figure 8 shows the how this variable varies. The average final number of features that remain in each top-level generalization is essentially independent of the number of features needed to create a generalization, but it does depend upon the number needed to retain a category. The lack of effect of the creation threshold is despite the fact that the initial number of features in a generalization, also shown in Figure 8, clearly does depend on that parameter.

26

```
Number of features in a top-level generalization
12                                11.4
11                                  i
10                                 I
 9        8.1           iI         10.8
 8         i         9.0
 7        IF           F            F
 6        7.4          6.7          7.1
 5         f            f            f
 4        5.5          5.5          5.5
 3    I -- pct to retain gen = 25% (initial features)
 2    i -- pct to retain gen = 20% (initial features)
 1    F -- pct to retain gen = 25% (final features)
 0    f -- pct to retain gen = 20% (final features)
     ----------------------------------------------
         25%          37.5%         50%
         Feature percent to generalize
```
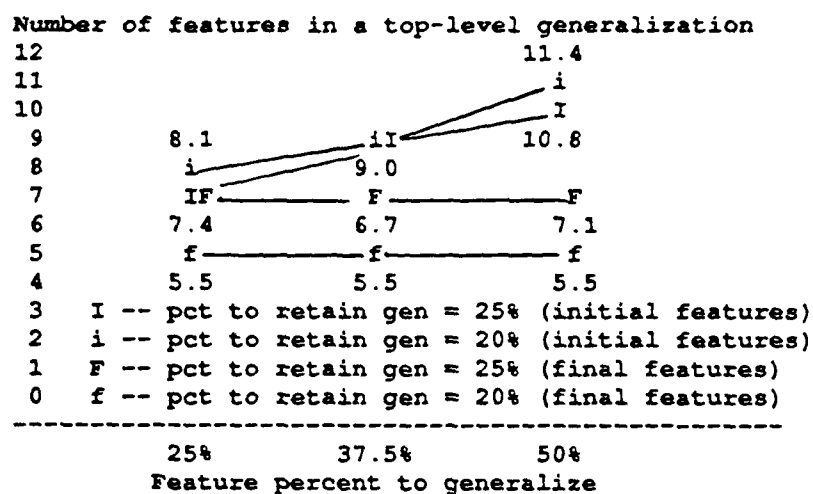
Figure 8: Average top level initial and final features as a function of
percentage to generalize and percentage to retain

Another one of our evaluation criteria was that, to the extent possible, instances should be evenly divided among the various concepts. To measure this, for each top-level generalization we counted the instances stored under it and its children. We looked at the standard deviation among the top-level generalizations, normalized in terms of the mean.[25] Figure 9 shows the results with this dependent measure. In general, the standard deviation increases (the instances are less well distributed) with the number of features needed to make a generalization. The (50%, 20%) point is a notable exception. This pair of parameter values also produced good behavior on the other measures as well. However, the results are not robust enough to draw any strong conclusions. In particular, the standard deviation results are very susceptible to new top-level generalizations created near the end of a run that describe only a small number of instances. We plan to examine at this situation in more detail using longer runs or by restricting our analysis to generalizations with high-confidence features.

The final dependent variable we considered was the average feature confidence level for top-level generalizations, which we use as an approximate measure of convergence. We present the results in Figure 10. The most notable thing about the data is that confidence levels are much higher for the 20%

---

[25]The normalization is needed since the total number of instances stored in the hierarchy can vary widely because instances can be "lost" when generalizations are deleted and because they can be stored in more than one place in memory.
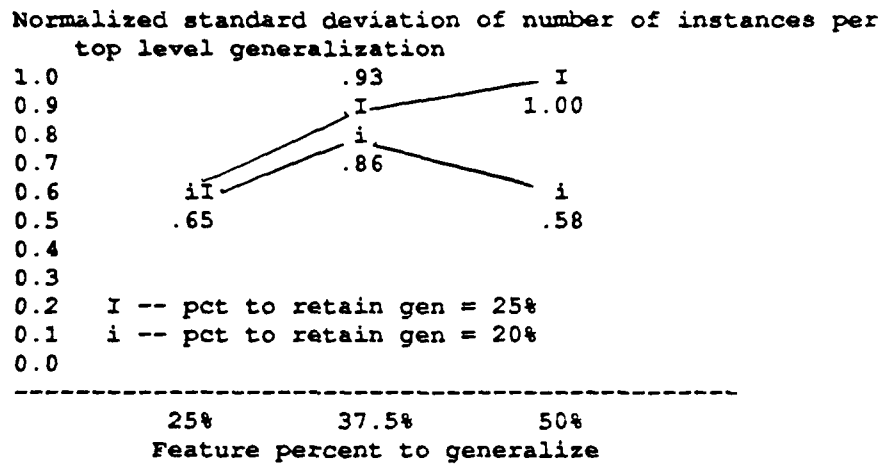
```
Normalized standard deviation of number of instances per
    top level generalization
1.0                          .93            I
0.9                      I                  1.00
0.8                      i
0.7                      .86
0.6            iI                           i
0.5            .65                          .58
0.4
0.3
0.2    I -- pct to retain gen = 25%
0.1    i -- pct to retain gen = 20%
0.0
     ------------------------------------------------
            25%          37.5%          50%
            Feature percent to generalize
```

Figure 9: Average instance distribution as a function of
percentage to generalize and percent to retain

retention level than for the 25% level and that the (50%, 20%) parameter combination clearly produces

the highest confidence levels. The lower levels at the 25% retention level probably resulted from the

failure of the concept set to converge before the run ended. Thus, the average reflects a number of

generalizations that would ultimately have been removed entirely. We need to examine longer runs to

determine whether this is strictly a convergence phenomenon or whether the confidence levels will remain

lower even with a fixed set of generalizations. UNIMEM typically will not converge upon a final set of

generalizations if one selects particularly poor parameter values. Whether this property is good or bad is

unclear.

```
Average feature confidence for top level generalizations
11                                  c
10                          c      11.2
 9                  c
 8      c          8.9
 7      8.2
 6
 5                      C           C
 4      C          5.4            4.8
 3      3.6
 2   C -- pct to retain gen = 25%
 1   c -- pct to retain gen = 20%
 0
   ------------------------------------------------
          25%          37.5%          50%
          Feature percent to generalize
```
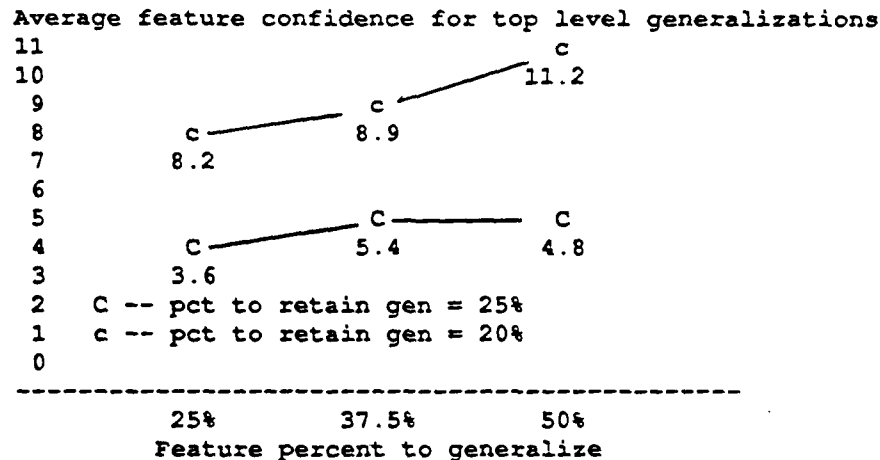
Figure 10: Average feature confidence as a function of
percentage to generalize and percentage to retain

The results in this section illustrate the various trade-offs involving the UNIMEM parameters. Considering first the percentage to retain a generalization parameter, the lower value (20%) produced generalizations with features that had higher confidence, a desirable result, indicating more rapid convergence. However, the higher value (25%), as expected, produced generalizations with more features, which is also desirable. The results involving numbers of generalizations produced were largely inconclusive with respect to this parameter, though they did indicate that the range of values we tried produced reasonable results in terms of our evaluation criteria.

Percentage of features needed to generalize, the other parameter under consideration, generally produced better results with the highest value (50%). The generalizations under that condition tended to have higher confidence features and more features and there were a reasonable number of generalizations. On the other hand, higher values of this parameter seemed to produce less well distributed instances, although the 50% value actually produced the best result in combination with the 20% value of the percentage to retain parameter. Notice that if we let the percentage to generalize parameter increase further and approach 100%, only identical instances would be used to create new concepts, which does not seem acceptable. Hence a trade-off is apparent.

It appears, then, that there is a need for intermediate values for each of the two parameters that we have examined. It seems that if instances are generalized on the basis of too few features then they are not necessarily very similar, and so their generalization has little predictive power. UNIMEM's confidence evaluation methods work well when a good generalization is embedded in the initial one, but not when the initial generalizations are essentially random. In contrast, if we require large numbers of features to generalize (especially larger values than the ones used here) then the initial generalized instances have so much in common that the generalization applies to few other instances and yet appears relevant to many of them. This undermines the ability of the confidence evaluation methods to identify irrelevant features. If too many features are required to retain generalizations in relation to the number needed to make them, then almost all of the generalizations will be disconfirmed. If too few are required, then the remaining generalizations become essentially meaningless.

While collecting the data for this experiment we also saved information about the computer time needed to incorporate instances into memory. UNIMEM was designed so that the time needed to add instances to memory should increase only slightly as memory grows. Indeed, if memory reaches a point where most of the new examples are duplicates of existing ones, and hence cause no changes to the concept hierarchy, then update time should be nearly constant. In any case, the tree structure of memory should result in the time needed to update memory growing at no more than a logarithmic rate, with the growth constant depending on how efficiently instances and sub-generalizations are indexed. Figure 11 shows the empirical results for one run in the university domain, averaged over groups of ten instances.[26] The growth of update time appears consistent with a logarithmic increase hypothesis and it clearly does not explode in any extreme way. We plan additional experiments to better estimate the growth rate and to examine UNIMEM's behavior when greater numbers of instances are involved.

```
Average seconds to incorporate
8
                                    -------      -------      -------
7                                  |7.281|      |7.474|      |7.542|
                                   |     |      |     |      |-----|     |
6                        ------|   |     |------|      |6.406|     |
                   ------|5.916|   |5.973|      |     |      |
5                 |5.371|     |     |     |      |     |     |
           ------|     |     |     |     |      |     |     |
4         |4.872|     |     |     |     |      |     |     |
    ------|     |     |     |     |     |      |     |     |
3   |3.325|     |     |     |     |     |      |     |     |
    |     |     |     |     |     |     |      |     |     |
2   |     |     |     |     |     |     |      |     |     |
 ------|     |     |     |     |     |     |      |     |     |
1 |1.452|     |     |     |     |     |     |      |     |     |
  |     |     |     |     |     |     |     |      |     |     |
0 ----------------------------------------------------------------
  1-10 11-20 21-30 31-40 41-50 51-60 61-70 71-80 81-90 91-100
            Number of instances in memory
```
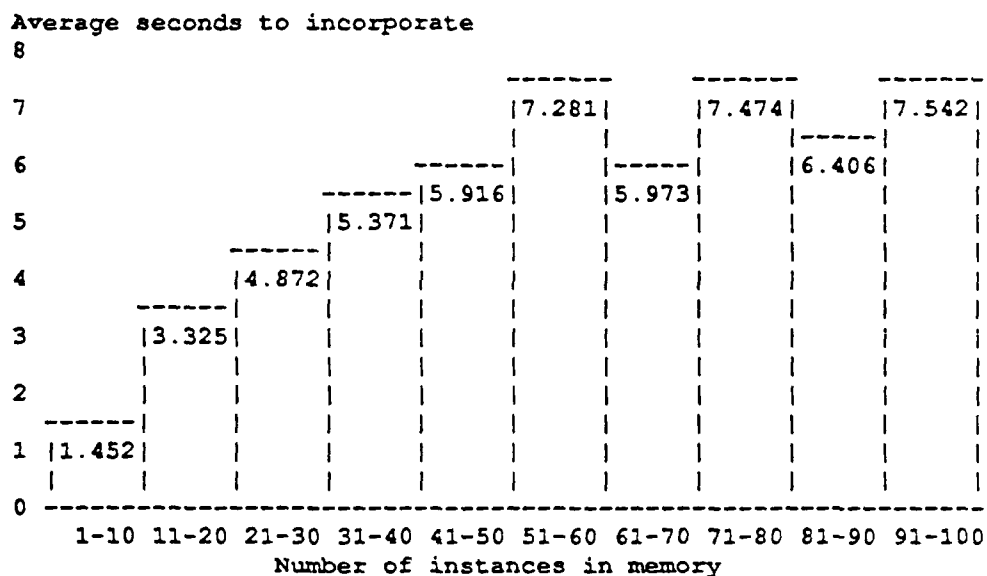
Figure 11: Time to incorporate new instances as a function of
instances already in memory

In conclusion, although we do not view the results presented in this section as definitive, they have given some insight into the effects of UNIMEM parameters. In addition, we feel that they show the kind of data that must be collected before we can fully understand the nature of learning by observation.

---

[26]We average the data over groups of ten instances since individual update times can vary radically depending upon whether a new generalization node needs to be created and how quickly the search through the concept hierarchy is narrowed.

4 Related research issues

Our work with UNIMEM has left us with a number of interesting problems to pursue. We briefly describe two of them here: the automatic modification of parameter settings and the integration of explanation-based methods with the empirical approach of UNIMEM.

4.1 Automatic setting of UNIMEM parameters

We have seen that UNIMEM uses many parameters and that their settings greatly affect the system's behavior. In the long run we would like the program to set these parameters itself for each new domain. The basic idea is that UNIMEM would monitor its behavior and adjust parameters to guide it toward the desired kind of generalization hierarchy. The goal would be expressed as another set of parameters, but ones that would make intuitive sense to a user, such as the rate at which generalizations are created, the rate they are deleted, or the average branching factor of the generalization tree.

It should not be difficult to extend UNIMEM in this fashion. The system would incrementally collect data about its behavior and periodically consider adjusting its parameters in response. However, we must first understand the effects of the various parameters through experiments of the sort described above. As an initial attempt at automatic parameter adjustment we plan to have UNIMEM monitor the rate at which generalizations are deleted, and if this rate becomes too high or too low have the system modify the parameters discussed in the previous section.

4.2 Using domain-dependent knowledge

Frequently when observing the world, humans attempt to to *explain* the generalizations that they make (Schank, 1986), an ability that is lacking in UNIMEM. We are currently studying the relationship between the empirical learning of the sort UNIMEM does and the *explanation-based* learning methods that have been developed recently (e.g., DeJong and Mooney, 1986, Mitchell et al., 1986, and Silver, 1986). These methods, instead of looking for regularities among a large number examples, analyze a single example in terms of cause and effect and generalize on the basis of this analysis. Roughly speaking, these methods explain an example and then generalize it, eliminating elements that are not essential to the explanation.

31

We have written elsewhere (Lebowitz, 1986a, 1986c) about the need to integrate these two styles of learning and our initial attempt to do so using UNIMEM. The basic approach is to explain empirically-produced generalizations using a simple rule base and to modify the UNIMEM generalizations where this is not possible. We have also begun a new project that focuses on the interaction between the two learning methods in understanding terrorism events (Danyluk, 1987).

Four assumptions underlie our plan for integrated learning. First, while an important goal of learning is indeed a causal model, and many explanation-based methods consider the causality behind examples, it is often not possible to determine underlying causality. Even where it is possible, it may not be computationally practical. Second, similarity usually indicates causality, is much easier to determine, and predictability can be used to help determine the direction of causality. Third, there exist methods to refine generalizations to mitigate the effects of coincidence, some of which we have seen in this paper. Finally, explanation-based and empirical methods complement each other effectively. In particular, it seems more efficient to use explanation-based methods to analyze generalizations rather than every individual example. Explanations can also help in deciding which empirical generalizations are likely to be significant and which features to consider.

5 Relation to other work

UNIMEM is closely related to Michalski and Stepp's (1983) conceptual clustering which was developed independently and contemporaneously with our work on Generalization-Based Memory. Fisher and Langley (1985) survey conceptual clustering methods. Michalski and Stepp's systems take instances represented much like those of UNIMEM and produce a hierarchical set of concept descriptions to describe them. The underlying mechanism, is, however, quite different. In particular, Michalski and Stepp's methods are not incremental, making use of an algorithm that looks at all of the instances at once. By making use of an algorithm that first finds maximally general discriminants and then determines maximally specific definitions of the resulting concepts, conceptual clustering aims to optimize the predictive power of concepts. Instances are required to be perfectly described by the concept that describes them, but this could presumably be relaxed for a performance element. Later versions of conceptual clustering (Stepp and Michalski, 1986) make use of domain goals to guide the search for descriptive concepts.

More recently, Fisher (1986) has developed COBWEB a system that, like UNIMEM, performs incremental concept formation from input instances represented with features. COBWEB shares with UNIMEM the general approach of top-down building of a concept hierarchy. A significant strength of Fisher's research is its use of an information measure developed by Gluck and Corter (1985) to determine an optimal concept division. This allows the work to be considered in theoretical as well as operational terms. COBWEB uses concept definitions are probabilistic (e.g., "50% of the instances in concept X have feature f"), unlike UNIMEM's simple conjunctive definitions. COBWEB has other differences in task definition -- it uses disjoint categories, and allows only nominal feature values -- but the general approach is similar. It would also appear that the information measure used by COBWEB will be more computationally expensive that UNIMEM's matching process, but confirming this would require a detailed comparison of computation times on similar domains with matched machines. However, the key point is that the growth rate for both systems is small, at worst logarithmic in the number of instances.

Another program carrying out the same task is Hanson and Bauer's (1986) WITT. Like COBWEB, WITT makes use of an information content metric, but unlike either COBWEB or UNIMEM it builds up its concept hierarchy bottom-up and is not incremental.

The research that is most closely related to the work described here is that of Kolodner (1984) with her CYRUS program. CYRUS was initially developed at Yale contemporaneously with IPP, UNIMEM's predecessor.[27] Like UNIMEM, CYRUS builds up hierarchies of generalizations based on similarities among instances. The primary difference is that CYRUS makes much more use of domain information, which it uses to determine which elements of instances can best serve as discriminants among concepts. CYRUS is able to handle instances that contain more information than do the UNIMEM instances since it can apply domain knowledge to avoid combinatorial explosions in search and concept formation. However, this also limits its flexibility in application to new domains. It is both a strength and weakness of UNIMEM that it relies on a feature representation that requires little domain information.

---

[27]Both IPP and CYRUS were heavily influenced by Schank's (1982) theory of memory organization packets (MOPs), which is a psychologically-oriented theory of memory that was under development at the same time as these programs. CYRUS also involved a major effort in intelligent question answering, including reconstructing information. Section 3.2 describes how UNIMEM differs from IPP.

UNIMEM, and indeed all the work in conceptual clustering, can also be compared to work in statistical clustering, e.g., Anderberg (1973). The way that instances are grouped by our methods bears considerable resemblance to the trees produced by hierarchical statistical clustering. These algorithms use as input a matrix of distances between pairs of instances. This can either be computed from a more complex representation (such as the feature representations that we used), or by evoking direct similarity measures (from human subjects, for example). The conceptual hierarchy is built solely from the similarity matrix by grouping together instances that are near to each other.

Unlike conceptual methods, statistical clustering algorithms do not form descriptions of the clusters that they find. This is a direct result of using only distance measures in the algorithm. This is quite acceptable if we have no other data. If we have more complex representations of instances, then conceptual methods can take advantage of it. In particular, statistical methods are subject to grouping together instances that are pairwise close, but have no common similarities as well as failing to group instances that have a core of similarity, but differ in other respects, so that they are not close together overall. In addition, the incremental nature of UNIMEM allows it to avoid computing distances between each pair of instances by only looking at those that are necessary. Conceptual clustering is further contrasted with statistical methods by Michalski and Stepp (1983). Appendix V shows a tree built from 47 university instances using a single linkage, farthest neighbor binary clustering algorithm the sum of feature differences (as defined for UNIMEM) as the similarity metric. Note that there are *not* any associated concept definitions.

6 Conclusion

Incremental concept formation is an important area of machine learning involving the automatic construction of a knowledge base that organizes real-world information. In this paper we have given an overview of UNIMEM, a program that performs concept formation incrementally. We demonstrated the system's generality by showing it in operation on several disparate domains. We also showed several of its key processes, including the creation of generalized concepts and their evaluation over time. We reported an experiment relating two of UNIMEM's parameters to the quality of the resulting set of concepts. While each new domain brings its own problems, the basic methods described here have

34

proven to be quite robust. We feel that UNIMEM constitutes a promising step along the way toward systems that can make maximal use of information that they collect over time.

Acknowledgments

I. A complete UNIMEM generalization hierarchy

Below we present the complete concept hierarchy formed in a run of UNIMEM on 224 university descriptions. The features that define each node are not presented. Figure 1 and Table 2 were taken from this hierarchy. The complete set of input is available from the author upon request.

```
GND0 [DALLAS-BAPTIST-COLLEGE JUILLIARD MICHIGAN-STATE SUNY-BUFFALO
      UNIVERSITY-OF-MISSISSIPPI VASSAR]
  GND1 [CCNY SUNY-BINGHAMTON UNIVERSITY-OF-NORTHCAROLINA YALE]
      GND26 [NORTHWESTERN REED]
          GND148 [TULANE UNIVERSITY-OF-PENNSYLVANIA]
          GND155 [GEORGE-WASHINGTON UNIVERSITY-OF-HARTFORD]
      GND44 [AUGSBURG OKLAHOMA-STATE-UNIVERSITY]
      GND63 [BARUCH UNIVERSITY-OF-MASSACHUSETTS-AMHERST
            WILLIAM-PATERSON-COLLEGE]
      GND72 [BAYLOR-UNIVERSITY TEXAS-CHRISTIAN-UNIVERSITY]
      GND73 [CONNECTICUT-COLLEGE GEORGE-WASHINGTON NORTHWESTERN]
          GND140 [CLARK-UNIVERSITY COLGATE]
          GND165 [ORAL-ROBERTS-UNIVERSITY TEXAS-CHRISTIAN-UNIVERSITY]
      GND76 [MIT PRINCETON]
      GND85 [UNIVERSITY-OF-CHICAGO UNIVERSITY-OF-NOTRE-DAME VANDERBILT]
      GND103 [NICHOLLS-STATE UNIVERSITY-OF-SOUTHDAKOTA]
          GND166 [ORAL-ROBERTS-UNIVERSITY WILLIAM-PATERSON-COLLEGE]
      GND114 [RICE SMITH SWARTHMORE]
      GND123 [COLGATE WESLEYAN]
          GND149 [PENN-STATE UNIVERSITY-OF-PENNSYLVANIA]
      GND124 [ILLINOIS-TECH OREGON-INSTITUTE-OF-TECHNOLOGY]
      GND126 [UNIVERSITY-OF-CALIFORNIA-SAN-DIEGO UNIVERSITY-WEST-VIRGINIA]
      GND128 [NYU TRINITY-COLLEGE]
      GND137 [SAN-JOSE-STATE UNIVERSITY-OF-LOWELL]
  GND2 [CHALMERS-UNIVERSITY-OF-TECHNOLOGY ECOLE-POLYTECHNIQUE PENN-STATE
        SAN-JOSE-STATE UNIVERSITY-OF-CALIFORNIA-SAN-DIEGO
        UNIVERSITY-OF-TEXAS]
      GND60 [UNIVERSITY-OF-COLORADO UNIVERSITY-OF-MASSACHUSETTS-AMHERST]
  GND3 [PENN-STATE SUNY-BINGHAMTON YALE]
      GND13 [BARD]
          GND156 [GEORGE-WASHINGTON LEWIS-AND-CLARK]
      GND15 [RUTGERS TRINITY-COLLEGE UNIVERSITY-OF-WASHINGTON]
      GND17 [HUNTINGTON-COLLEGE MESA ORAL-ROBERTS-UNIVERSITY
            UNIVERSITY-OF-PORTLAND]
      GND31 [UNIVERSITY-OF-PUGET-SOUND]
          GND109 [REED SMITH]
          GND115 [MANHATTANVILLE-COLLEGE SWARTHMORE]
          GND142 [CONNECTICUT-COLLEGE TRINITY-COLLEGE]
      GND41 [CLARK-UNIVERSITY STEVENS WASHINGTON-AND-LEE]
      GND64 [BARUCH UNIVERSITY-OF-MASSACHUSETTS-AMHERST
            WILLIAM-PATERSON-COLLEGE]
      GND81 [TEXAS-CHRISTIAN-UNIVERSITY TOURO]
          GND157 [COLGATE GEORGE-WASHINGTON]

(more)
```

Figure 12: A complete UNIMEM generalization hierarchy

```
        GND88 [NORTHCAROLINA-STATE-UNIVERSITY UNIVERSITY-OF-LOWELL
               UNIVERSITY-OF-NORTHCAROLINA]
        GND90 [ROCHESTER-TECH UNIVERSITY-OF-NOTRE-DAME
               UNIVERSITY-OF-PENNSYLVANIA]
        GND104 [NICHOLLS-STATE UNIVERSITY-OF-SOUTHDAKOTA]
        GND118 [PRINCETON WESLEYAN]
        GND125 [TULANE UNIVERSITY-OF-CHICAGO]
        GND127 [UNIVERSITY-OF-CALIFORNIA-SAN-DIEGO UNIVERSITY-OF-DENVER]
        GND131 [QUEENS SAN-JOSE-STATE]
        GND152 [NORTHWESTERN NYU]
   GND4 []
        GND9 [HARVARD UNIVERSITY-OF-PENNSYLVANIA]
            GND119 [COLUMBIA WESLEYAN]
        GND19 [MIT SWARTHMORE]
            GND133 [PRINCETON YALE]
            GND134 [BROWN YALE]
        GND20 [BRANDEIS BROWN]
        GND27 [COLGATE CONNECTICUT-COLLEGE LEWIS-AND-CLARK RICE]
            GND116 [REED SWARTHMORE]
            GND129 [TRINITY-COLLEGE WESLEYAN]
        GND33 [CLARK-UNIVERSITY MANHATTANVILLE-COLLEGE]
        GND39 [BAYLOR-UNIVERSITY UNIVERSITY-OF-TULSA]
        GND74 [STEVENS TEXAS-CHRISTIAN-UNIVERSITY]
        GND82 [BOSTON-UNIVERSITY TOURO]
        GND86 [UNIVERSITY-OF-CHICAGO UNIVERSITY-OF-NOTRE-DAME VANDERBILT]
        GND97 [NYU ROCHESTER-TECH]
        GND110 [SMITH TRINITY-COLLEGE WASHINGTON-AND-LEE]
        GND153 [CORNELL NORTHWESTERN]
        GND154 [NORTHWESTERN TUFTS]
        GND167 [AUGSBURG ORAL-ROBERTS-UNIVERSITY]
   GND5 [ORAL-ROBERTS-UNIVERSITY RICE]
        GND21 [MIT SWARTHMORE]
            GND135 [PRINCETON YALE]
            GND136 [BROWN YALE]
        GND28 [NEWENGLAND-COLLEGE REED]
        GND34 [CLARK-UNIVERSITY MANHATTANVILLE-COLLEGE]
        GND35 [CLARK-UNIVERSITY LEHIGH-UNIVERSITY]
        GND49 [BOSTON-UNIVERSITY STEVENS]
        GND52 [BAYLOR-UNIVERSITY UNIVERSITY-OF-PUGET-SOUND]
        GND68 [COLGATE NEWYORKIT NORTHWESTERN UNIVERSITY-OF-PORTLAND]
            GND150 [CORNELL UNIVERSITY-OF-PENNSYLVANIA]
        GND83 [TEXAS-CHRISTIAN-UNIVERSITY TOURO]
        GND87 [UNIVERSITY-OF-NOTRE-DAME VANDERBILT]
        GND111 [SMITH TRINITY-COLLEGE WASHINGTON-AND-LEE]
        GND120 [UNIVERSITY-OF-CHICAGO UNIVERSITY-OF-PENNSYLVANIA]
            GND141 [COLGATE WESLEYAN]
        GND143 [CONNECTICUT-COLLEGE TULANE]
        GND158 [GEORGE-WASHINGTON NYU]
```

(more)

Figure 12, continued

```
GND8 [CCNY ILLINOIS-TECH NORTHCAROLINA-STATE-UNIVERSITY PRINCETON
        SUNY-BINGHAMTON UNIVERSITY-OF-CALIFORNIA-SAN-DIEGO]
    GND14 [CONNECTICUT-COLLEGE UNIVERSITY-OF-ROCHESTER]
        GND159 [COLGATE GEORGE-WASHINGTON]
        GND160 [GEORGE-WASHINGTON LEWIS-AND-CLARK]
    GND23 [NYU SUNY-ALBANY TEXAS-CHRISTIAN-UNIVERSITY
            UNIVERSITY-OF-NORTHCAROLINA UNIVERSITY-OF-WASHINGTON
            WASHINGTON-AND-LEE]
        GND53 [BOSTON-UNIVERSITY UNIVERSITY-OF-PUGET-SOUND]
        GND161 [GEORGE-WASHINGTON VANDERBILT]
    GND24 [SUNY-ALBANY SUNY-PURCHASE]
        GND162 [COLGATE GEORGE-WASHINGTON]
        GND163 [GEORGE-WASHINGTON UNIVERSITY-OF-PUGET-SOUND]
    GND32 [CONNECTICUT-COLLEGE ORAL-ROBERTS-UNIVERSITY RICE]
        GND84 [MANHATTANVILLE-COLLEGE TOURO]
        GND112 [REED SMITH]
        GND121 [COLGATE SWARTHMORE UNIVERSITY-OF-CHICAGO]
        GND130 [TRINITY-COLLEGE UNIVERSITY-OF-PUGET-SOUND]
    GND40 [UNIVERSITY-OF-PENNSYLVANIA UNIVERSITY-OF-TULSA]
        GND138 [BAYLOR-UNIVERSITY UNIVERSITY-OF-LOWELL]
        GND164 [GEORGE-WASHINGTON UNIVERSITY-OF-PUGET-SOUND]
    GND42 [CLARK-UNIVERSITY STEVENS WASHINGTON-AND-LEE]
    GND55 [TEXAS-A&M UNIVERSITY-OF-OKLAHOMA UNIVERSITY-WEST-VIRGINIA]
    GND93 [NORTHWESTERN ROCHESTER-TECH]
        GND144 [CONNECTICUT-COLLEGE YALE]
        GND145 [CONNECTICUT-COLLEGE TRINITY-COLLEGE]
        GND151 [UNIVERSITY-OF-NOTRE-DAME UNIVERSITY-OF-PENNSYLVANIA]
    GND105 [NICHOLLS-STATE UNIVERSITY-OF-SOUTHDAKOTA]
        GND168 [ORAL-ROBERTS-UNIVERSITY WILLIAM-PATERSON-COLLEGE]
GND7 [CAL-TECH ORAL-ROBERTS-UNIVERSITY OREGON-INSTITUTE-OF-TECHNOLOGY]
    GND45 [AUGSBURG CORPUS-CHRISTI-STATE-U]
    GND69 [BRYN-MAWR UNIVERSITY-OF-PORTLAND]
    GND100 [MORGAN-STATE UNIVERSITY-OF-DENVER]
    GND106 [NICHOLLS-STATE UNIVERSITY-OF-SOUTHDAKOTA]
    GND117 [RICE SMITH SWARTHMORE]
    GND146 [CONNECTICUT-COLLEGE TRINITY-COLLEGE]
GND10 [ORAL-ROBERTS-UNIVERSITY TOURO WASHINGTON-AND-LEE]
    GND29 [BRYN-MAWR REED]
    GND36 [CLARK-UNIVERSITY MANHATTANVILLE-COLLEGE]
    GND47 [AUGSBURG HUNTINGTON-COLLEGE]
    GND48 [AUGSBURG WALLA-WALLA-COLLEGE]
    GND70 [STEVENS UNIVERSITY-OF-PORTLAND]
    GND113 [COLORADO-COLLEGE SMITH]
    GND147 [CONNECTICUT-COLLEGE TRINITY-COLLEGE]
GND16 [GEORGE-WASHINGTON GOTHENBURG-UNIVERSITY NORTHWESTERN
        ORAL-ROBERTS-UNIVERSITY WASHINGTON-AND-LEE]
    GND58 [NEWYORKIT UNIVERSITY-OF-PUGET-SOUND]
    GND59 [NEWYORKIT SAM-HOUSTON-STATE-UNIVERSITY]
    GND75 [TEXAS-CHRISTIAN-UNIVERSITY UNIVERSITY-OF-PORTLAND]
    GND80 [BARUCH UNIVERSITY-OF-TOLEDO]
    GND98 [NYU TUFTS]
    GND108 [NICHOLLS-STATE UNIVERSITY-OF-SOUTHDAKOTA]
    GND139 [SAN-JOSE-STATE WILLIAM-PATERSON-COLLEGE]
```

Figure 12, completed

II. UNIMEM parameters

What follows is a complete list of the parameters used to control the UNIMEM generalization process. Omitted are parameters used only to control the form of the output. Typical values are given in parentheses.

- Percentage of similar instance features needed to create a generalization (40%).
- Absolute minimum number of features needed to generalize (2).
- Percentage of instance features needed to keep a generalization after some features have been deleted (20%).
- Absolute minimum number of features needed to keep a generalization (2).
- Total amount of conflict between instance features and generalization features allowed in search (1.0).
- Confidence level at which a feature is deleted (-3).
- Confidence level at which a feature is presumed permanent (20).
- Distance apart feature values can be and still be assumed to match (0.5).
- Confidence multiplier for matches (2.0).
- Confidence multiplier for mismatches (-2.0).
- Number of features that indicate relevance in search (2).
- Number of misses, less than which indicates relevance (2).
- Penalty for missing feature (0.1).

III. Instances used for the example in Section 2.3

Attribute	Value for PRINCETON	Value for HARVARD	Value for MIT
STATE	NEW-JERSEY	MASSACHUSETTS	MASSACHUSETTS
LOCATION	SMALL-TOWN	URBAN	URBAN
CONTROL	PRIVATE	PRIVATE	PRIVATE
MALE:FEMALE	65:35	65:35	75:25
NO-OF-STUDENTS	< 5,000	5,000-10,000	< 5,000
STUDENT:FACULTY	7:1	10:1	5:1
SAT-VERBAL	650	700	650
SAT-MATH	675	675	750
EXPENSES	> $10,000	> $10,000	> $10,000
%-FINANCIAL-AID	50	60	50
NO-APPLICANTS	10,000-13,000	13,000-17,000	4,000-7,000
%-ADMITTANCE	20	20	30
%-ENROLLED	60	80	60
ACADEMICS	5 out of 5	5 out of 5	5 out of 5
SOCIAL	3 out of 5	3 out of 5	3 out of 5
QUALITY-OF-LIFE	3 out of 5	4 out of 5	3 out of 5
ACAD-EMPHASIS	HISTORY	HISTORY	SCIENCE
ACAD-EMPHASIS	ECONOMICS	BIOLOGY	ELEC-ENGINEERING
ACAD-EMPHASIS	POLITICAL-SCIENCE	LIBERAL-ARTS	MECH-ENGINEERING
ACAD-EMPHASIS	LIBERAL-ARTS		ENGINEERING
ACAD-EMPHASIS	ENGINEERING		

Attribute	Value for CASE-WESTERN	Value for AUBURN	Value for ARIZONA-STATE
STATE	OHIO	ALABAMA	ARIZONA
LOCATION	URBAN	SMALL-TOWN	
CONTROL	PRIVATE	STATE	STATE
MALE:FEMALE	70:30	11:9	50:50
NO-OF-STUDENTS	< 5,000	15,000-20,000	> 20,000
STUDENT:FACULTY	9:1	18:1	20:1
SAT-VERBAL	550	480	450
SAT-MATH	650	545	500
EXPENSES	> $10,000	< $4,000	> $4,000-7,000
%-FINANCIAL-AID	65	50	50
NO-APPLICANTS	< 4,000	4,000-7,000	> 17,000
%-ADMITTANCE	85	90	80
%-ENROLLED	35	60	60
ACADEMICS	3 out of 5	2 out of 5	3 out of 5
SOCIAL	2 out of 5	4 out of 5	4 out of 5
QUALITY-OF-LIFE	3 out of 5	4 out of 5	5 out of 5
ACAD-EMPHASIS	ENGINEERING	EDUCATION	BUSINESS-EDUCATION
ACAD-EMPHASIS	MANAGEMENT	BUSINESS-ADMIN	ACCOUNTING
ACAD-EMPHASIS	ARTS-AND-SCIENCES	ENGINEERING	FINE-ARTS
ACAD-EMPHASIS		HEALTH-SCIENCE	ENGINEERING
ACAD-EMPHASIS		PRE-PROFESSIONAL	
ACAD-EMPHASIS		SOCIAL-SCIENCE	

Table 6: Six more universities

IV. Configuring UNIMEM to process terrorist stories

In addition to the comparison of IPP-MEM and UNIMEM behavior, we present our work with the terrorism domain in order to illustrate some of the issues that arise in trying use real-world input for a machine learning program. In order to experiment with the terrorist event domain, we had to construct an interface that translated the output of the natural language module of IPP (IPP-NLP) into a set of features (as was originally done for IPP-MEM) which were given to UNIMEM as input. This configuration is illustrated in Figure 13. UNIMEM's memory can be used in a performance fashion by IPP-NLP for some default inferences. If a new instances matches with a generalization in memory, but has no values for some features contained in the generalization, then those features are added to the description of the instance.

```
 ----------------                    --------------------              ----------
|IPP-NLP text| =>       text      => |   representation | => feature => | UNIMEM |
| processing |    representation     | post-processing  |    list        ----------
 ----------------                    --------------------
```
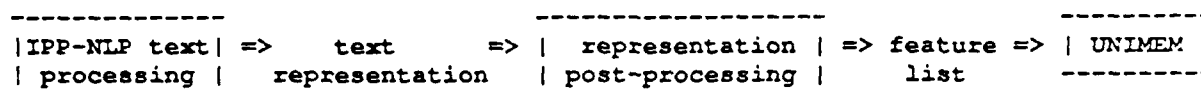
Figure 13: IPP-NLP/UNIMEM

The output of IPP-NLP was prepared for either IPP-MEM or UNIMEM (which received identical input) by producing features about the action (what it was -- hijacking, bombing, etc.; where it was; when it was) and about each of the role fillers (actor, weapon, etc.). For each role filler, features were taken from modifiers in the text. Other features, such as that a soldier is in the military, were inferred using a simple frame-based memory.[28] Table 7 shows a typical, if simple, terrorism story and the features that were generated for it from the output of IPP-NLP. There are both direct translations of adjectives (e.g., "right wing" in the text becomes the feature actor-politics/conservative) and inferred features (e.g., target-place-type/public).

---

[28]The inferred features were necessary since certain facts, such as that soldiers are in the military, are so basic that stories will never mention them. The addition of such features by the interface does involve a clear bias in the sense of Utgoff (1986).

41

New York Times (UPI), 12 December 1979

A plastic bomb apparently set by right-wing extremists exploded early today
in the doorway of the town hall of Lezo in the northern Basque region.

```
Attribute              Value
------------------------------------------------------------------
METHODS                $EXPLODE-BOMB          [the first group of
LOCATION-AREA          WESTERN-EUROPE          features describe
LOCATION-NATION        *SPAIN*                 the event as a whole]
S-MOP                  S-DESTRUCTIVE-ATTACK
TIME                   EARLY
ACTOR-POLITICS         CONSERVATIVE           [the rest are from the
ACTOR-POL-POS          ACTIVIST                various role fillers]
ACTOR-NATIONALITY      *SPAIN*
WEAPON-CLASS           EXPLOSIVE
WEAPON-COMPOSITION     PLASTIC
WEAPON-WEAPON          BOMB
TARGET-PLACE-TYPE      PUBLIC
TARGET-DIRECTION       NORTH
TARGET-NATIONALITY     *BASQUE*
TARGET-PLACE           LEZO
TARGET-NATIONALITY     *SPAIN*
TARGET-PLACE           TOWN-HALL
TARGET-PLACE-TYPE      GOVERNMENT
TARGET-PLACE           DOORWAY
```

Table 7: Typical terrorist event features provided to UNIMEM

V. Statistical clustering on university instances

This tree was built from 47 university instances using a single linkage,
farthest neighbor binary clustering algorithm. The similarity measure
between instances was the sum of feature differences, exactly as defined for
UNIMEM. There are *no* associated concept definitions. Lengths of the
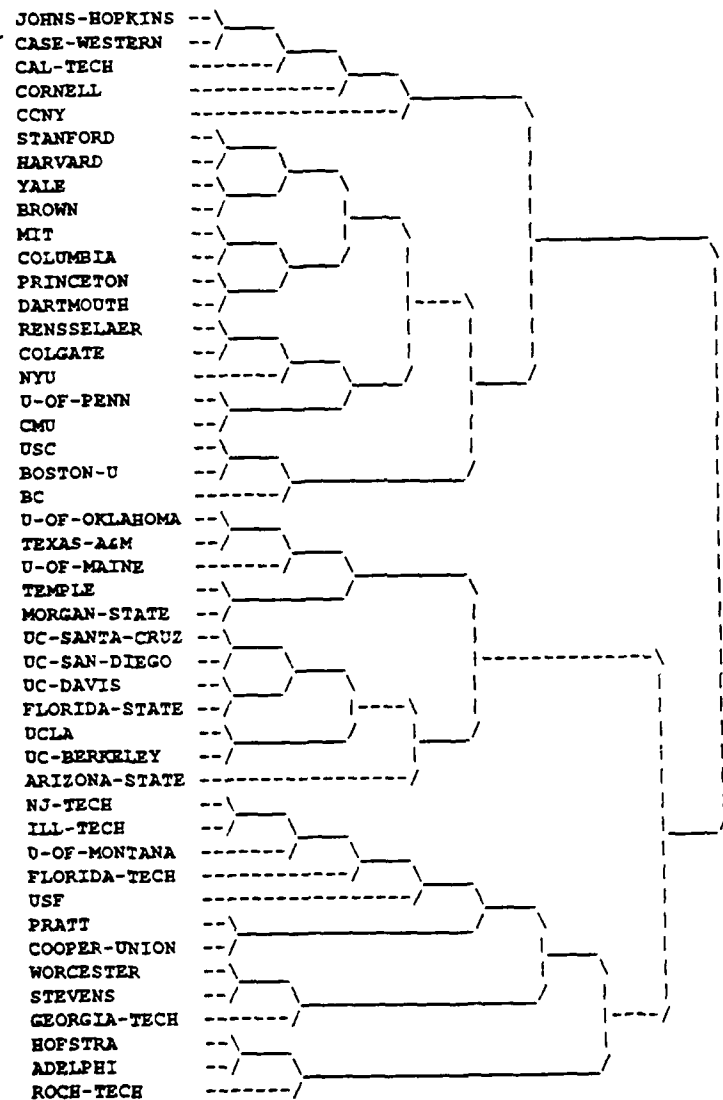edges are not significant.

```
JOHNS-HOPKINS --\
CASE-WESTERN  --/    \
CAL-TECH      -------/    \
CORNELL       ------------/    \
CCNY          ------------------/    \
STANFORD      --\                     |
HARVARD       --/    \                |
YALE          --\     \               |
BROWN         --/      \              |
MIT           --\       \     \       |
COLUMBIA      --/    \    \     \      \
PRINCETON     --\     \    |     |      |
DARTMOUTH     --/      |---\|     |     |
RENSSELAER    --\      |    |     |     |
COLGATE       --/      |    |     |     |
NYU           -------/    \  /    |     |
U-OF-PENN     --\         \ /     |     |
CMU           --/          |      |     |
USC           --\          |      |     |
BOSTON-U      --/    \_____/      |     |
BC            -------/            |
U-OF-OKLAHOMA --\                 |
TEXAS-A&M     --/    \            |
U-OF-MAINE    -------/    \       |
TEMPLE        --\         \       |
MORGAN-STATE  --/         |       |
UC-SANTA-CRUZ --\         |       |
UC-SAN-DIEGO  --/    \    |----------------\  |
UC-DAVIS      --\     \   |               |  |
FLORIDA-STATE --/    |----\               |  |
UCLA          --\____/    |   |            |  |
UC-BERKELEY   --/         |   |            |  |
ARIZONA-STATE ------------------/         |  |
NJ-TECH       --\                         |  |
ILL-TECH      --/    \                    |__/
U-OF-MONTANA  -------/    \               |
FLORIDA-TECH  ------------/    \          |
USF           ------------------/    \    |
PRATT         --\                     \   |
COOPER-UNION  --/                      \  |
WORCESTER     --\                       \ |
STEVENS       --/    \              /    \|
GEORGIA-TECH  -------/             |----/
HOFSTRA       --\                  |
ADELPHI       --/    _____/
ROCH-TECH     -------/
```

Figure 14: Statistical clustered universities

References

Anderberg, M. R. (1973). *Cluster Analysis for Applications.* New York: Academic Press.

Brachman, R. J. (1985). I lied about the trees. *AI Magazine, 6,* 80 - 93.

Charniak E., Riesbeck, C. K., and McDermott, D. V. (1980). *Artificial Intelligence Programming.* Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Danyluk, A. P. (1987). *The use of explanations for similarity-based learning.* Columbia University Department of Computer Science.

DeJong, G. F. and Mooney, R. (1986). Explanation-based learning: An alternative view. *Machine Learning, 1,* 145 - 176.

Dietterich, T. G. and Michalski, R. S. (1986). Learning to predict sequences. In R. S. Michalski, J. G. Carbonell and T. M. Mitchell, (Ed.), *Machine Learning: An Artificial Intelligence Approach, Volume II,* (pp. 63 - 106). Los Altos, CA: Morgan Kaufmann.

Feigenbaum, E. A. (1963). The simulation of verbal learning behavior. In E. A. Feigenbaum and J. Feldman, (Ed.), *Computers and Thought,* (pp. 297 - 309). New York: McGraw-Hill.

Fisher, D. (1986). *The acquisition of basic level concepts via incremental conceptual clustering.* Irvine Computational Intelligence Project.

Fisher, D. and Langley, P. (1985). Approaches to conceptual clustering. *Proceedings of the Ninth International Joint Conference on Artificial Intelligence* (pp. 691 - 697). Los Angeles: Morgan Kaufmann.

Gluck, M. and Corter, J. (1985). Information, uncertainty, and the utility of categories. *Proceedings of the Seventh Annual Conference of the Cognitive Science Society* (pp. 283 - 287). Irvine, CA.

Hanson, S. J. and Bauer, M. (1986). Conceptual clustering, semantic organization and polymorphy. *Proceedings of the International Meeting on Advances in Learning* (pp. 53 - 77). Les Arc, France.

Kolodner, J. L. (1984). *Retrieval and Organizational Strategies in Conceptual Memory: A Computer Model.* Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Lebowitz, M. (1980). *Generalization and memory in an integrated understanding system.* (Tech. Rep. No. 186). New Haven, CT: Yale University Department of Computer Science.

Lebowitz, M. (1983). RESEARCHER: An overview. *Proceedings of the Third National Conference on Artificial Intelligence* (pp. 232 - 235). Washington, DC: Morgan Kaufmann.

Lebowitz, M. (1983). Generalization from natural language text. *Cognitive Science, 7,* 1 - 40.

Lebowitz, M. (1983). Memory-based parsing. *Artificial Intelligence, 21,* 363 - 404.

Lebowitz, M. (1985). Classifying numeric information for generalization. *Cognitive Science, 9,* 285 - 308.

Lebowitz, M. (1986). Not the path to perdition: The utility of similarity-based learning. *Proceedings of the Fifth National Conference on Artificial Intelligence* (pp. 533 - 537). Philadelphia, PA: Morgan Kaufmann.

Lebowitz, M. (1986). An experiment in intelligent information systems: RESEARCHER. In R. Davies, (Ed.), *Intelligent Library and Information Systems,* (pp. 127 - 149). London: Ellis Horwood.

Lebowitz, M. (1986). Integrated learning: Controlling explanation. *Cognitive Science, 10,* 219 - 240.

Michalski, R. S. and Stepp, R. E. (1983). Automated construction of classifications: Conceptual clustering versus numerical taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 5,* 396 - 409.

Minsky, M. (1975). A framework for representing knowledge. In P. H. Winston, (Ed.), *The Psychology of Computer Vision.* New York: McGraw-Hill.

Mitchell, T. M. (1982). Generalization as search. *Artificial Intelligence, 18,* 203 - 226.

Mitchell, T. M., Keller, R. M. and Kedar-Cabelli, S. T. (1986). Explanation-based generalization: A unifying view. *Machine Learning, 1,* 47 - 80.

Quillian, M. R. (1968). Semantic memory. In M. Minsky, (Ed.), *Semantic Information Processing.* Cambridge, MA: MIT Press.

Schank, R. C. (1982). *Dynamic Memory: A Theory of Reminding and Learning in Computers and People.* New York: Cambridge University Press.

Schank, R. C. (1986). *Explanation Patterns.* Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Silver B. (1986). Precondition analysis: Learning control information. In R. S. Michalski, J. G. Carbonell and T. M. Mitchell, (Ed.), *Machine Learning: An Artificial Intelligence Approach, Volume II,* (pp. 647 - 670). Los Altos, CA: Morgan Kaufmann.

Stepp, R. E. and Michalski, R. S. (1986). Conceptual clustering: Inventing goal-oriented classifications of structured objects. In R. S. Michalski, J. G. Carbonell and T. M. Mitchell, (Ed.), *Machine Learning: An Artificial Intelligence Approach, Volume II,* (pp. 471 - 498). Los Altos, CA: Morgan Kaufmann.

Utgoff, P. E. (1986). *Machine Learning of Inductive Bias.* Norwell, MA: Kluwer Academic.

Winston, P. H. (1972). Learning structural descriptions from examples. In P. H. Winston, (Ed.), *The Psychology of Computer Vision,* (pp. 157 - 209). New York: McGraw-Hill.